UIUC in HARD 2004 – Passage Retrieval Using HMMs

Jing Jiang ChengXiang Zhai
Department of Computer Science
University of Illinois at Urbana-Champaign

Abstract

UIUC participated in the HARD track in TREC 2004 and focused on the evaluation of a new method for identifying variable-length passages using HMMs. Most existing approaches to passage retrieval rely on pre-segmentation of documents, but the optimal boundaries of a relevant passage depends on both the query and the document. Our new method aims at determining or improving the boundaries of a relevant passage based on both the query and topical coherence in the document. In this paper, we describe the method and present analysis of our HARD 2004 evaluation results. The results show that the HMM method can improve the boundaries of pre-segmented passages in terms of overall passage retrieval accuracy and recall, but at the price of precision sometimes. However, due to the non-optimality of the relevance feedback procedure and the poor ranking performance based on passage scoring, the best of our passage runs is still worse than a whole document baseline run. Further experiments and analysis are needed to fully understand why the language modeling approach did not work well on passage scoring.

1 Introduction

Most information retrieval systems return a ranked list of whole documents as answers to a query. However, when documents are long and have multiple topics, retrieval at passage-level, i.e., returning relevant passages, rather than whole documents, may be more useful to the user as the user does not need to read through a whole document to find the most relevant part. Passage retrieval also enables an IR system to re-score documents based on their relevant passages, and exploit feedback more accurately based on passages rather than whole documents. Indeed, previous work [9, 1, 12, 3, 6, 4] has shown that retrieval performance can be improved by using passage-level evidence.

Current passage retrieval methods usually pre-segment documents into passages of fixed length [11]. A disadvantage of such kind of methods is that the passage length is not adaptive to specific query and specific document. However, as we would expect, the length of the most relevant passage in a document depends both on the specific query and the document itself. Therefore, ideally, a passage retrieval method should be able to retrieve *variable-length* arbitrary passages from documents.

The difficulty of variable-length passage retrieval lies on the large search space: for a document of n-word long, there are $O(n^2)$ possible passages to consider. It is not practical to treat each of these passages as an individual document and rank them. A method based on *Hidden Markov Models* (*HMMs*) can tackle this problem by using dynamic programming techniques. In this method, a document is not modeled as a bag of words but as a sequence of words generated from a probabilistic model that involves transitions between a finite number of states. Baum-Welch algorithm is used to set appropriate parameters in the HMM, and Viterbi algorithm is used to detect the most relevant passage in a document. The advantages of the HMM-based passage retrieval methods are that the passage can start from and end at any arbitrary word in the document, and that the parameters that control the passage length can be trained.

Mittendorf and Schäuble first proposed to use HMMs for passage retrieval [7]. However, their work focused on using passage retrieval to improve document ranking rather than to accurately detect the passage boundary as we will explore. Moreover, they mapped words to a different domain to capture the similarity between each word and the query, and used numbers in this domain as output symbols of the HMM. The mapping inevitably introduces additional computation and heuristic parameters, which we avoid by using the words in a document directly as output symbols. Denoyer *et al.* also used HMMs to identify relevant passages in whole documents, but their passages were of fixed length, and they focused on using scores of passages for document classification and ranking [2].

HARD track in TREC 2004 provided a good test bed for evaluation of our HMM-based passage retrieval

method. Twenty-five out of the fifty evaluation topics had retrieval element set at the passage-level. We tested our HMM-based passage retrieval method on these 25 topics. A pre-segmentation-based passage retrieval method was also used as a baseline for our evaluation.

The paper is organized as follows. In Section 2, we introduce our HMM-based passage retrieval method in details. We then describe our HARD 2004 experiment setup in Section 4 and discuss the results in Section 5. Finally, we conclude with Section 6.

2 HMM-Based Passage Retrieval

In this section, we first briefly review Hidden Markov Models, then describe the proposed HMM-based passage retrieval method in details.

2.1 HMM Basics

A first order Hidden Markov Model (HMM) defines a discrete stochastic process that generates a sequence of output symbols from a sequence of hidden states. State transition occurs according to some transition probabilities, and output generation occurs according to some output probabilities. Formally, a first order HMM consists of the following components:

1. A set of hidden states $S = \{s_1, \ldots, s_n\}$.
2. A set of observable output symbols $O = \{o_1, \ldots, o_m\}$.
3. An initial probability $a_{0,i}$ for each state s_i. $a_{0,i}$ is the probability that a state sequence starts from state s_i. $\sum_{i=1}^{n} a_{0,i} = 1$.
4. A transition probability $a_{i,j}$ for each pair of states (s_i, s_j). $a_{i,j}$ is the probability that the next state is s_j given that the current state is s_i. $\sum_{j=1}^{n} a_{i,j} = 1$ for $i = 1, \ldots, n$.
5. An output probability $b_{i,k}$ for each pair of a state and an output symbol (s_i, o_k). $b_{i,k}$ is the probability that symbol o_k is generated from state s_i. $\sum_{k=1}^{m} b_{i,k} = 1$ for $i = 1, \ldots, n$.

Given an HMM with all the parameters listed above specified, the probability of getting a state sequence $s_{p_1} s_{p_2} \ldots s_{p_T}$, where s_{p_t} is the state at time t, is $\prod_{t=1}^{T} a_{p_{t-1}, p_t}$. Given that we know the state sequence is $s_{p_1} s_{p_2} \ldots s_{p_T}$, the probability of generating an output sequence $o_{v_1} o_{v_2} \ldots o_{v_T}$, where o_{v_t} is the output symbol at time t, is $\prod_{t=1}^{T} b_{p_t, v_t}$. If we are only given the observed output sequence $o_{v_1} o_{v_2} \ldots o_{v_T}$ without knowing the underlying state sequence, the most likely state sequence that may have generated this output sequence can be efficiently computed using Viterbi algorithm, a dynamic programming algorithm. Given an HMM with some or all parameters (the probabilities) unspecified, these parameter values can be estimated based on some observed sequences of symbols with or without their corresponding underlying state sequences. Baum-Welch algorithm (essentially an Expectation-Maximization algorithm) provides an efficient way for unsupervised training of HMMs. A detailed tutorial on HMMs is given in [8].

2.2 An HMM Approach to Passage Retrieval

Our basic idea of applying HMMs to passage retrieval is as follows. The set of all words in a document collection forms the set of output symbols of the HMM. A document is seen as a sequence of words (output symbols). A set of hidden states generate these words. Each state has its own word distribution, or what we call a language model. Each state is either relevant or non-relevant to the query. By decoding the document, we get a most likely state sequence that has generated the sequence of words. Words generated from those hidden states that are relevant to the query form the relevant passage from the original document. As we can see, the boundary of the relevant passage is automatically detected by the HMM. With such a method for locating the relevant passage in a long, relevant document, we can perform passage retrieval in two ways: (1) first rank documents using any IR method, then extract a relevant passage from each top-ranked document, and (2) first extract possibly relevant passages from documents, then rank the documents based on the relevant passages they contain. In HARD 2004, we only explored the first strategy.

2.3 Choices of HMMs

The simplest HMM for passage retrieval consists of 3 states connected linearly, as shown in Figure 1. The first and the third states are non-relevant states, or what we call background states. They generate words according to a background language model. The second state is relevant to the query, and it generates words according to some relevant language model. This model assumes that a document contains a single passage relevant to the query. A document is generated by starting with some non-relevant part, then switching to the relevant passage, and finally switching back to some non-relevant part.

We could use the collection language model as the background language model for states B1 and B2. The relevant language model for state R cannot be obtained easily. Certainly it is related to the query. But simply using query language model is not enough because a relevant passage contains many non-query words. Without relevance feedback, the best way is to smooth this language model with a background language model.

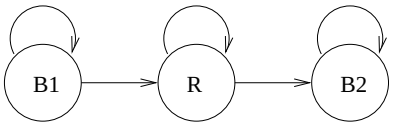

Figure 1: 3-State HMM

An improvement of the 3-state HMM is a 4-state HMM as shown in Figure 2, where a last state that only generates a special end-of-document symbol is added to the system. Our previous experiments on a different data set showed that adding this special state can improve the performance.

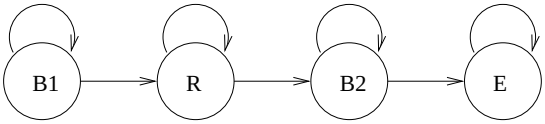

Figure 2: 4-State HMM

A disadvantage of the 3-state and the 4-state HMMs is that the smoothing factor needs to be fixed. Smoothing can also be done automatically within the HMM if we separate the query language model and the background language model at R. Figure 3 shows the modified HMM. State B1 and state B3 are considered non-relevant, while state R and state B2 are considered relevant. Non-query words in the relevant passage are now generated from B2 rather than from R, as B2 uses the background language model. Thus, smoothing is achieved through the transitions between R and B2. An advantage of this model is that the smoothing factor, which is the transition probability between R and B2 in this case, can be estimated through training rather than being heuristically set.

To incorporate feedback in this system, we could use the feedback language model at state R. This modification gives us the final HMM we used in our HARD 2004 experiments. This HMM is shown in Figure 4.

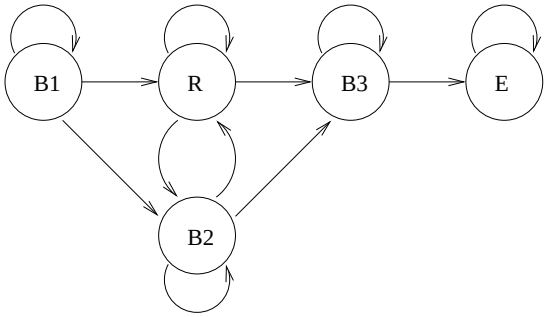

Figure 3: 5-State HMM

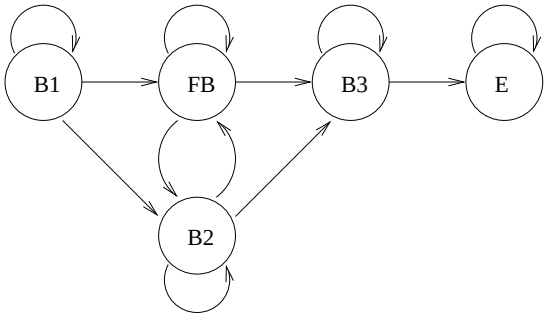

Figure 4: 5-State HMM with Feedback

3 Preliminary Experiments on HARD 2003

Prior to trying our HMM-based method on HARD 2004 data, we did some preliminary experiments on HARD 2003 data, mainly to test if HMM-based method can indeed detect variable-length passages better than fixed-length passage retrieval methods.

We set up the experiments on HARD 2003 data as follows. We first extracted out the set of whole documents that were judged to contain relevant passages to some topic. This information was obtained according to the passage-level judgment file for HARD 2003. For each of these documents and its corresponding relevant topic, we used two baseline methods and our HMM method without pseudo-feedback (Figure 3) to extract the relevant passage. The first baseline method is quite simple: it extracts the passage that starts from the first occurrence of a query word, and ends at the last occurrence of a query word. This method is straightforward and computationally cheap. The second baseline method we used is stronger: it uses a fixed-length sliding window to scan the whole document, and picks the passage that has the most

	Simple Baseline	Strong Baseline (different window sizes)			HMM
		200	400	600	
Prec	0.631	0.525	0.491	0.476	0.525
Rec	0.705	0.692	0.888	0.941	0.976
F1	0.516	0.508	0.543	0.542	0.587

Table 1: Results from Simple Baseline, Strong Baseline, and HMM on HARD 2003

		Prec	Rec	F1
Simple Baseline	w/o FB	0.631	0.705	0.516
	w/ FB	0.542	0.975	0.604
Strong Baseline (200)	w/o FB	0.525	0.692	0.508
	w/ FB	0.497	0.987	0.568
Strong Baseline (400)	w/o FB	0.491	0.888	0.543
	w/ FB	0.481	0.992	0.552
Strong Baseline (600)	w/o FB	0.476	0.941	0.542
	w/ FB	0.473	0.993	0.542
HMM	w/o FB	0.610	0.749	0.532
	w FB	0.525	0.976	0.587

Table 2: Comparison of Performance before and after Pseudo Feedback on HARD 2003 data

occurrences of query words among all passages of the fixed length. Passages extracted by this method can start at arbitrary places in documents, but have fixed length.

We compared passage extracted from these methods with the true passages indicated in the judgment file. We used three measures to evaluate the performance. Precision is the number of words in the overlapping part between the extracted passage and the true passage divided by the total number of words in the extracted passage. Recall is the number of words in the overlapping part between the extracted passage and the true passage divided by the total number of words in the true passage. F1 is a harmonic mean of the precision and recall. Table 1 shows the performance measures of the two baseline methods and the HMM method.

We then used these extracted passages for pseudo-feedback in our HMM-based method, as illustrated in Figure 4. The feedback language model is constructed from the previously extracted passage using either one of the baseline method or the simple HMM method without feedback. Performance is shown in Table 2.

Our conclusions from our experiments on HARD 2003 and from our other experiments on a different data set are that (1) HMM method using pseudo-feedback from a simple method that has comparative high precision (such as our simple method) gives better overall performance (measured by F1) than our baseline methods, and that (2) HMM method performs consistently well over a range of passage lengths, while fixed-length passage retrieval method cannot handle variable-length passages well.

4 HARD 2004 Experiment Setup

In this section, we describe our passage retrieval experiments in HARD 2004. As the metadata "retrieval-element" was not available for baseline runs, only our final runs explored passage retrieval methods.

4.1 Baseline Run and Clarification Forms

For baseline run, we used Lemur toolkit to retrieve a ranked list of whole documents from the corpus for each topic. We used K-L divergence language model retrieval method [5, 13] and pseudo-feedback with 5 documents. This baseline run ("uiucHARDb0") turned out to be our best run.

For clarification forms, we presented to the user 6 documents for each topic. These were the gapped top-6 documents with gap set to 3. The gapped top-k is a simple heuristic method proposed in [10] to increase the diversity of the documents presented to the user. As 6 whole documents cannot fit in one screen, we used HMM method to retrieve a passage from each of these documents, and presented to the user the first 50 and the last 50 words of each passage.

4.2 Final Runs – Passage Retrieval

First, we used the relevance feedback from the clarification forms to expand the query models of the 50 evaluation topics. For topics with retrieval-element set to document, we used the K-L divergence method with the expanded query models to retrieve top 1000 whole documents from the corpus. There results were returned in the finals runs "uiucHARDf0" and "uiucHARDf1", but only

for the 25 topics that had "retrieval-element" set to "document".

For passage retrieval, we used two methods in our final runs. In the first method, each document was first pre-segmented into non-overlapping passages, each containing 120 words. We chose 120 as the passage length as this was the average passage length from HARD 2003 passage-level judgments. We then treated each fixed-length passage as an individual document, and retrieved top 1000 passages for each topic, using the K-L divergence method. There results were returned in our final run "uiucHARDf0".

We used the 5-state HMM with pseudo-feedback for passage retrieval in our second final run. The pseudo-feedback was obtained as follows. We first pre-segmented all documents into non-overlapping passages of 60-word long. We then used the K-L divergence method to retrieve top 1000 passages from these 60-word long passages for each topic. For each retrieved passage, we applied the HMM-based method on the document where this passage was extracted from. We used the 60-word long passage to construct a feedback language model for the 5-state HMM. The variable-length passage returned by the HMM was then used to replace the original 60-word long passage. There passages were returned in our final run "uiucHARDf1" for the 25 passage-level topics. Essentially, we were using a 60-word fixed length passage as a "seed" passage to train the feedback language model in the 5-state HMM and extract a new variable-length passage from the same document. Our hope was that HMM could refine the boundary of the original 60-word long passage, which is largely confirmed in our experiment results.

We chose 60 as the length of the pre-segmented passages because shorter passages usually have higher precision (but lower recall) than longer passages. From our previous experiments, we learned that having a high precision of the feedback language model in the 5-state HMM is more important than having a high recall.

5 HARD 2004 Experiment Results

5.1 Document-Level Results

Forty-five out of the original fifty topics were judged at the document-level, as the other five topics did not have any relevant documents. Our baseline run "uiucHARDb0" returned whole documents for all topics. Our final runs "uiucHARDf0" and "uiucHARDf1" returned whole documents to the 25 topics that are at document-level, and passages to the other 25 topics that are at passage-level.

Run	R Prec (Hard)	R Prec (Soft-Hard)
uiucHARDb0	0.3574	0.3325
uiucHARDf0	0.2834	0.3145
uiucHARDf1	0.2690	0.3015

Table 3: Document-Level Average Precision (All Topics)

Run	R Prec (Hard)	R Prec (Soft-Hard)
uiucHARDb0	0.3590	0.3567
uiucHARDf0	0.3219	0.3575
uiucHARDf1	0.3219	0.3555

Table 4: Document-Level Average Precision (Doc-Level Topics Only)

Table 3 shows the overall average precision for each run on all topics. Table 4 shows the precision over only the document-level topics for each run.

We see that baseline run b0 performed the best among these three runs, though the Soft-Hard R-precision for document-level topics is essentially the same for all the three runs. From Table 4, we see that the baseline run uiucHARDb0, which is a pseudo feedback run on whole document index, somehow ranks documents more accurately than the relevance feedback runs on pre-segmented passage indices (i.e., uiucHARDf0 and uiucHARDf1), suggesting that either our relevance feedback procedure is not quite effective or the KL-divergence method does not work well with short passages.

We were expecting that the relevance feedback runs would outperform the baseline run because the final runs incorporated relevance feedback. But the evaluation of the results shows that the relevance feedback we exploited did not improve the performance much. Thus we looked into the relevance feedback procedure and discovered that some parameter setting is apparently non-optimal, causing very conservative feedback, which may at least partially explain the relatively poor performance of relevance feedback. Some follow-up experiments indicate that our relevance feedback (reflected in the trained query model based on relevance judgments) does perform better than without any feedback, but it somehow does not perform as well as pseudo feedback. In addition to the non-optimal parameter setting, another possible reason may be that the user did not judge any document, or judged only one document to be relevant for some topics, in which case we have very limited information for feedback, whereas the pseudo feedback always uses the top 5 documents.

Comparing Table 3 and Table 4, we see that while

the baseline whole document scoring run has roughly the same Hard R-precisions in both tables, the two feedback passage scoring runs have significantly worse R-precisions on passage-level topics, which is worth further examination.

5.2 Passage-Level Results

We are more interested in the passage-level evaluation as our main focus was on passage-level retrieval. As our final run f0 used a fixed passage length of 120 words, but our HMM-based final run f1 was based on fixed-length passages of 60 words, in order to compare the HMM-based method and the pre-segmentation-based method and see the effect of using HMMs to improve the boundaries of a pre-segmented passage, we evaluated another two runs using the scripts provided by NIST; they correspond to using pre-segmented fixed-length passages, with a fixed-length of 60 and 120 words, respectively.

Table 5 compares the fixed length passage results (for both 60 words and 120 words) with the corresponding HMM runs on several different performance measures. To avoid any complication caused by multiple, potentially overlapping passages, we filtered the results so that there is at most one passage from each document (i.e., each document contributes at most one passage). As a result, the absolute performance is lower than that with the complete results. But since our official HMM results (uiucHARDf1) are filtered, we also filtered other results to make the results completely comparable.

The results show that for both 60-word passages and 120-word passages, the HMM method outperforms the corresponding pre-fixed baseline by character-based measures. In particular, the HMM runs have better performance in both cases in terms of BPref@12K characters, which is the recommended major measure for passage retrieval. By passage-based measures, the HMM runs are better in recall and F values, but worse in precision. Based on these observations, we may conclude that HMMs can improve the boundaries of fixed-length passages mostly by increasing the recall and sometimes decreasing the precision, which is as we expected.

Table 6 shows the passage-level performance of our three official runs. We see that although the HMMs generally improve the recall and combined measures such as BPrec@12K over the fixed length passages, the official run uiucHARDf1 (Fixed60+HMM) is worse than uiucHARDf0 (Fixed120), suggesting that the improvement from using HMMs is not that much as guessing the right length of relevant passages. A somehow surprising observation is that the baseline uiucHARDb0 performs much better than both relevance feedback runs in BPref@12K and recall at 10 passages, though it performs worse in precision at 10 passages. To understand why, we looked into specific differences between uiucHARDb0 and uiucHARDf1 and evaluated the performance of each component. First, we looked at returning whole documents as passages. Ranking of the documents can be based on original queries without feedback, with pseudo feedback, or with relevance feedback. We then looked at returning whole documents as passages but ranking the documents based on the score of the best fixed-length passage from each document. Finally we compared those runs with uiucHARDf1, which retrieved passages and was based on fixed-length passage scoring. To ensure that all components are comparable, we truncate the results so that the evaluation is all performed only on the top 400 documents. The results are shown in Table 7.

From this table, we can make several interesting observations:

1. Pseudo feedback (with top 5 documents) improves performance across all measures.

2. Relevance feedback (with gapped top-k) only improves performance by passage-based measures but *decreases* performance substantially for character-based measures. This may be because the relevant documents we obtained from the user are mostly down on the list (due to the use of gapped top-k), and using them for feedback may not help improve the front-end precision which is what the character-based measures emphasize.

3. Comparison between relevance feedback and pseudo feedback indicates that the character-based measures appear to favor high recall.

4. Passage-based scoring is generally much worse than whole-document scoring by all measures; 60-word passages are significantly worse than 120-word passages. This definitely needs further examination. One possible reason may be the KL-divergence method is not robust for scoring short passages.

5. Applying HMMs on top of 60-word passages slightly improves the performance, but the performance improvement is insufficient to balance the loss of performance due to the passage scoring.

6. Three factors have contributed to the large difference in BPref@12K between uiucHARDb0 and uiucHARDf1: (1) Relevance feedback performs worse than pseudo feedback. (2) Passage scoring

Method	BPref@12K	Prec@12K	CharRPrec	Rec@10Psg	Prec@10Psg	F@10Psg	PsgRPrec
Fixed60	0.1208	0.1623	0.0776	0.0412	0.169	0.0289	0.1154
Fixed60+HMM	0.1868	0.2143	0.1424	0.1494	0.1411	0.0706	0.1037
Fixed120	0.1738	0.2088	0.1043	0.0924	0.2056	0.051	0.133
Fixed120+HMM	0.2131	0.2265	0.1562	0.1698	0.1822	0.0764	0.1094

Table 5: "One passage pre document" evaluation of HMMs

Run	BPref 12000 Chars	Recall 10 Passages	Prec 10 Passages
uiucHARDb0	0.2710	0.2517	0.1570
uiucHARDf0	0.2080	0.1067	0.2391
uiucHARDf1	0.1860	0.1494	0.1411

Table 6: Passage-level performance of 3 official runs

is worse than whole document scoring. (3) Scoring with 60-word passages is worse than scoring with 120-word passages. The first two appear to be the dominating factors.

6 Conclusions

In this paper, we reported UIUC's TREC 2004 Hard Track experiments and results. We focused on the study of a new HMM-based method for identifying variable-length relevant passages from documents. The basic idea of this method is to model a document as a sequence of words generated from an HMM, which has two kinds of states – relevant states and background states. One critical problem with constructing such an HMM is to estimate the output probability from a relevant state (i.e., a unigram language model or word distribution). In our HARD 2004 experiments, we used a fixed-length passage obtained through pre-segmentation to estimate this relevance language model. The idea is essentially to use HMMs on top of a fixed-length passage to improve the boundaries of the fixed-length passage.

We evaluated the HMMs for passages of two different sizes (60 words and 120 words). Overall, the results show that the HMMs can improve the passage retrieval performance over fixed-length passages, mostly by increasing recall and thus some combined measures. This is consistent with what we observed in our preliminary experiments. However, a few other factors, e.g., relevance feedback, scoring whole documents vs. scoring passages, appear to be more dominant in determining the performance. Thus even though the HMM can improve performance over fixed length passages, our best passage retrieval performance is still much worse than our baseline performance.

Further experiments are needed to clarify issues such as the effectiveness of using the KL-divergence method for scoring passages.

References

[1] J. P. Callan. Passage-level evidence in document retrieval. In *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 302–310, 1994.

[2] L. Denoyer, H. Zaragoza, and P. Gallinari. HMM-based passage models for document classification and ranking. In *23rd European Colloquium on Information Retrieval Research*, 2001.

[3] M. Kaszkiel and J. Zobel. Passage retrieval revisited. In *Proceedings of the 20th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 178–185, 1997.

[4] M. Kaszkiel and J. Zobel. Effective ranking with arbitrary passages. *Journal of the American Society for Information Science*, 52(4):344–364, 2001.

[5] J. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of 24th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, 2001.

Measure	NoFB	PseudoFB	RelFB	RelFBPsg120	RelFBPsg60	RelFBPsg60HMM
BPref@12K	0.2671	0.2705	0.2417	0.1980	0.1700	0.1868
Prec@12K	0.2974	0.2989	0.2653	0.2137	0.1943	0.2143
CharRPrec	0.1912	0.1964	0.1883	0.1492	0.1294	0.1424
Rec@10Psg	0.2005	0.2517	0.2493	0.1843	0.1572	0.1494
Prec@10Psg	0.1387	0.157	0.1746	0.1648	0.1213	0.1411
F@10Psg	0.0892	0.1012	0.0999	0.0761	0.0684	0.0706
PsgRPrec	0.1074	0.1108	0.1204	0.0987	0.0926	0.1037

Table 7: Performance breakdown for top 400 documents

[6] X. Liu and B. Croft. Passage retrieval based on language models. In *Proceedings of the 11th International Conference on Information and Knowledge Management*, pages 375–382, 2002.

[7] E. Mittendorf and P. Schäuble. Document and passage retrieval based on hidden Markov models. In *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 318–327, 1994.

[8] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of IEEE*, 77:257–286, 1989.

[9] G. Salton, J. Allan, and C. Buckley. Approaches to passage retrieval in full text information systems. In *Proceedings of the 16th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 49–58, 1993.

[10] X. Shen and C. Zhai. Active feedback – UIUC trec-2003 HARD experiments. In *Proceedings of the 12th Text REtrieval Conference*, 2003.

[11] S. Tellex, B. Katz, J. Lin, A. Fernandes, and G. Marton. Quantitative evaluation of passage retrieval algorithms for question answering. In *Proceedings of the 26th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 41–47, 2003.

[12] R. Wilkinson. Effective retrieval of structured documents. In *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 311–317, 1994.

[13] C. Zhai and J. Lafferty. Model-based feedback in K-L divergence retrieval model. In *Proceedings of the 10th Internationl Conference on Information and Knowledge Management*, 2001.