Document Retrieval in the Context of Question Answering

Christof Monz

Language & Inference Technology, University of Amsterdam,
Nieuwe Achtergracht 166, 1018 WV Amsterdam, The Netherlands.
E-mail: christof@science.uva.nl
URL: www.science.uva.nl/~christof

Abstract. Current question answering systems rely on document retrieval as a means of providing documents which are likely to contain an answer to a user's question. A question answering system heavily depends on the effectiveness of a retrieval system: If a retrieval system fails to find any relevant documents for a question, further processing steps to extract an answer will inevitably fail, too. In this paper, we compare the effectiveness of some common retrieval techniques with respect to their usefulness for question answering.

1 Introduction

Document retrieval systems aim to return relevant documents to a user's query, where the query is a set of keywords. A document is considered relevant if its content is related to the query. Question answering (QA) systems, on the other hand, aim to return an (exact) answer to a question.

Since 1999, the annual Text REtrieval Conference (TREC) organized by the National Institute of Standards and Technology (NIST) features a question answering track. Given a large number of newspaper and newswire articles, participating systems try to answer a set of questions by analyzing the documents in the collection in a fully automated way.

Most, if not all, current question answering systems first use a document retrieval system to identify documents that are likely to contain an answer to the question posed. This pre-processing step, also referred to as pre-fetching, is mainly motivated by feasibility considerations. Question answering requires a deeper analysis of the documents, e.g., syntactic parsing, synonym linking, pattern matching, etc. It is impossible to do this for a complete collection of documents of reasonable size in an efficient manner. Therefore document retrieval is used to restrict the whole collection to a subset of documents which are probable to contain an answer, and then the actual process of answer selection is carried out on this subset.

The information needs for ad-hoc retrieval and document retrieval as a pre-fetch for question answering are quite different, viz. finding documents that are on the same topic as a query and documents actually containing an answer to a question, The issue at this point is whether techniques that have proved to be effective for ad-hoc document retrieval are equally effective for retrieval as pre-fetching for QA.

The importance of this questions lies in the strong impact of the effectiveness of a document retrieval system on the overall performance of the answer selection module: If a retrieval system does not find any relevant documents for some question, even a perfect answer selection module will not be able to return a correct answer. The PRISE retrieval system [9] was used by NIST (for TREC-10 and TREC-11) to provide participants in the QA track with potentially relevant documents, in case a participating group did not have a retrieval system. For example, using a cut-off of 20, which is in the vicinity of the cut-offs used by many participants in TREC QA tracks, PRISE failed to return any relevant documents for 28% of the questions of the TREC-11 data set. This affected not only questions which can be considered difficult by the current state of the art in QA, or questions which did not have an answer in the collection, but also relatively 'easy' questions such as (1) and (2).

(1) *What year did South Dakota become a state?* (topic id: 1467)
(2) *When was Lyndon B. Johnson born?* (topic id: 1473)

Our objective is to investigate what retrieval techniques allow for an optimization of document retrieval when used as a pre-fetch for QA.

To the best of our knowledge, there is hardly any systematic evaluation of document retrieval as pre-fetching for question answering. Which is somewhat surprising considering the number of QA systems employing document retrieval in one form or another. The only work focusing on this issue is [6], where the impact of passage-based retrieval vs. full document retrieval as pre-fetching is investigated.

The remainder of this paper is organized as follows: The next section explains the test data and retrieval techniques that are investigated. Section 3 presents the results of the experiments. Finally, section 4 gives some conclusions.

2 Experimental Setup

2.1 Test Data

We used the TREC-9, TREC-10, and TREC-11 data sets consisting of 500 questions each with 978,952 documents for TREC-9 and TREC-10 from the TIPSTER/TREC distribution and 1,033,461 documents for TREC-11 from the AQUAINT distribution. At TREC-9 and TREC-10, participants were required to return up to five answer-document-id pairs for each question, where the answer can be any text string containing maximally 50 characters, and the document-id refers to the document from which the answer was extracted. At TREC-11, participants were required to return one answer-document-id pair for each question, where the answer had to be the exact answer.

In addition, we used the judgment files which were provided by NIST as a result of their evaluation. A judgment file, which is comparable to a qrel file in ad-hoc retrieval, indicates for each submitted answer-document-id pair, whether the answer is correct and whether the document supports, i.e., justifies, the answer. The justifying documents form the set of relevant documents against which we evaluate the different document retrieval approaches for pre-fetching. If none of the participants returned a supported answer, that topic was discarded from our evaluation. This also included questions that did not have an answer in the collection, which can be the case since TREC-10.

The final evaluation sets consist of 480, 433, and 455 topics for TREC-9, TREC-10, and TREC-11, respectively. The original question set for TREC-9 actually contained 693 questions where 193 questions were syntactic variants of 54 of the remaining 500 questions. Here, we did not use the variants, but if a relevant document for a variant was included in the judgment file, it was added to the set of relevant documents of the original question. Variants were removed to avoid repetition of topics, which could bias the overall evaluation. We also included 10 topics of the TREC-11 question set, where, although none of the participants found a relevant document, NIST assessors 'coincidentally' recognized a document containing an answer during their evaluation.

One of the traits of the question answering data sets, compared to earlier ad-hoc retrieval data sets, is the much smaller number of relevant or supporting documents. Table 1 displays the statistical distribution of relevant documents over several data sets. As will be seen later on, this property does affect retrieval performance.

	TREC-4 ah	TREC-7 ah	TREC-8 ah	TREC-9 qa	TREC-10 qa	TREC-11 qa
median	74.0	55.0	68.5	7.0	5.0	3.0
mad	89.2	62.8	60.1	8.9	6.6	3.0

Table 1. The median number of relevant documents and the corresponding median absolute deviation (mad).

2.2 Document Retrieval Approaches

All retrieval techniques discussed in the remainder of this article use the FlexIR retrieval system [7]. FlexIR is a vector-space retrieval system with several features including positional indexing, blind feedback, and structured querying.

In this subsection we introduce some techniques which are known to have a positive impact on the effectiveness of document retrieval, and which have also been used by participants in TREC's question answering tracks. Of course, this is only a selection of retrieval techniques that can and have been applied to pre-fetching. Nevertheless, we aim to discuss some techniques that are commonly used.

Stemming. Stemming has a long tradition in document retrieval, and a variety of stemmers are available, see [3] for an overview. Here, we use the Porter stemmer [8], which is probably the most commonly used stemmer. Since the Porter stemmer is purely rule-based, it sometimes fails to recognize variants, e.g. irregular verbs such as *thought*, which is stemmed as *thought*. Therefore, we decided to also use a lexical-based stemmer, or lemmatizer [10]. Each word is assigned its syntactic root through lexical look-up. Mainly number, case, and tense information is removed, leaving other morphological derivations such as nominalization intact.

Some QA systems do not use stemming to avoid compromising early precision [2], while others use a hybrid approach where the index contains both, the original word and its stem, and matching the stem contributes less to the document similarity score than matching the original word.

Blind Relevance Feedback. Blind relevance feedback analyzes the top n (usually $5 \leq n \leq 10$) documents from a preliminary retrieval run to add new terms, and to reweight terms that were part of the original query. Blind feedback has become a standard technique in document retrieval because of its consistent and strong positive impact on retrieval effectiveness, cf. [11]. On the other hand it is not used in the context of question answering, which might be because there is only a small number of relevant documents, see table 1, and it is known that blind feedback performs rather poorly under those circumstances. Nevertheless, we wanted to confirm this empirically in the context of question answering. Our blind relevance feedback approach uses the top 10 documents and term weights were recomputed by using the standard Rocchio method. We allowed at most 20 terms to be added to the original query.

Passage-Based Retrieval. Passage-based retrieval splits a document into several passages, where passages can be of fixed length or vary in length, start at any position or at fixed positions, and overlap to a certain degree, see [4] for a comprehensive overview. Passage-based retrieval has proved particularly useful for document collections that contain longer documents, such as the Federal Register sub-collection of TREC. Using passages instead of whole documents emphasizes that the information sought by a user can be expressed very locally. This probably also explains its appeal to question answering, where answers tend to be found in a sentence or two, and it is not surprising that many QA systems use passage-based retrieval instead of document retrieval.

From the broad spectrum of available passage-based retrieval techniques, we used the approach described in [1], where all passages are of fixed length and each passage starts at the middle of the previous one. The first passage of a document starts with the first occurrence of a matching term. Given a query q and a document d which is split into passages $pass_d^1, \ldots, pass_d^n$, the similarity between q and d ($sim(q,d)$) is defined as $max_{1 \leq i \leq n} sim(q, pass_d^i)$. This mapping of passages to their original documents is mainly for evaluation purposes, as the NIST judgments are made with respect to document ids. When using a passage-based retrieval system in the context of an actual QA system one would probably like to return passages instead, as this allows the answer selection procedure to analyze smaller and more focused text segments.

3 Experimental Results

3.1 Stemming

The first retrieval technique we investigated is stemming. In the literature stemming is sometimes described as recall-enhancing, e.g., [5], and the question is whether retrieval as a pre-fetch to a question answering system can profit from stemming, in particular, since pre-fetching should opt for early precision. Table 2 shows the a@n scores for lower cut-offs, where a@n is the number of questions with at least one relevant document up to rank n.

One can notice that the improvements for TREC-10 are much lower than for the other two collections. This could be due to the much larger portion of definition questions in the TREC-10 question set. Questions asking for a definition often contain foreign or

	TREC-9		TREC-10		TREC-11	
a@*n*	lemma	+porter	lemma	+porter	lemma	+porter
a@5	0.6687	0.7000 (+4.6%)	0.6443	0.6490 (+0.7%)	0.4813	0.5231 (+8.6%)
a@10	0.7396	0.7854 (+6.1%)	0.7298	0.7344 (+0.6%)	0.6066	0.6264 (+3.2%)
a@20	0.8042	0.8458 (+5.1%)	0.7875	0.8014 (+1.7%)	0.6659	0.7055 (+5.9%)
a@50	0.8729	0.9146 (+4.7%)	0.8568	0.8753 (+2.1%)	0.7516	0.7956 (+5.8%)

Table 2. Comparison of the ratios of questions with at least one relevant document (a@n) using lemmas vs. porter stemming.

technical terms, see (3), or proper names, see (4), where in both cases morphological normalization does not apply very well, if at all.

(3) *What is amitriptyline?* (topic id: 936)
(4) *Who was Abraham Lincoln?* (topic id: 959)

Summing up, one can say that applying stemming consistently improves a@*n* scores, although the extent depends on the question type (e.g., definition questions show lower improvements) and the specificity of the question, i.e., if there is only a small number of documents containing an answer. For these reasons, and because stemming has become a standard technique in document retrieval, stemming is applied to all experiments discussed below, including the Lnu.ltc baseline run.

3.2 Blind Relevance Feedback

Similar to stemming, blind feedback has become an established technique in ad-hoc document retrieval throughout the years. The experimental results for blind feedback compared to plain retrieval are shown in table 3.

	TREC-9		TREC-10		TREC-11	
a@*n*	Lnu.ltc	+feedback	Lnu.ltc	+feedback	Lnu.ltc	+feedback
a@5	0.7000	0.6125 (−12.4%)	0.6490	0.5289 (−18.4%)	0.5231	0.4000 (−23.5%)
a@10	0.7854	0.7125 (−9.2%)	0.7298	0.6028 (−17.3%)	0.6264	0.4923 (−21.4%)
a@20	0.8458	0.7833 (−7.3%)	0.7875	0.7067 (−10.2%)	0.7055	0.5824 (−17.4%)
a@50	0.9146	0.8604 (−5.9%)	0.8568	0.8199 (−4.3%)	0.7956	0.7077 (−11.0%)

Table 3. Comparing simple and blind feedback retrieval.

feedback is not appropriate in the context of question answering. All runs dramatically decrease in performance. The bad performance of feedback is most likely due to the small number of relevant documents per topic. This could also explain why the results decrease from TREC-9 to TREC-11, as also the average number of relevant documents decreases, see table 1.

3.3 Passage-Based Retrieval

Passage-based retrieval is widely used in QA systems and is therefore worth analyzing in more detail. As mentioned in section 2.2, we chose to define passages in terms of

windows, where each window is of fixed length and overlaps 50% with the previous one. Defining windows this way, exhibited rather consistent improvements in earlier work on ad-hoc retrieval [1]. We experimented with 11 different window sizes: 10, 20, 30, 50, 70, 100, 150, 200, 250, 350, and 500 words. In all cases, the overlap ratio of 50% remained fixed.

The similarity between a query and passage was computed with the Lnx.ltc weighting scheme, which is similar to the Lnu.ltc weighting scheme except that document length normalization is not applied. Normalization was left out because all passages are of fixed length and therefore normalization is expected to make little difference.

Figure 1, shows the a@n scores for the three TREC collections, with $n \in \{5, 10, 20, 50\}$. In addition to the passage-based runs, also the results for the base runs, using full-document retrieval, are shown.

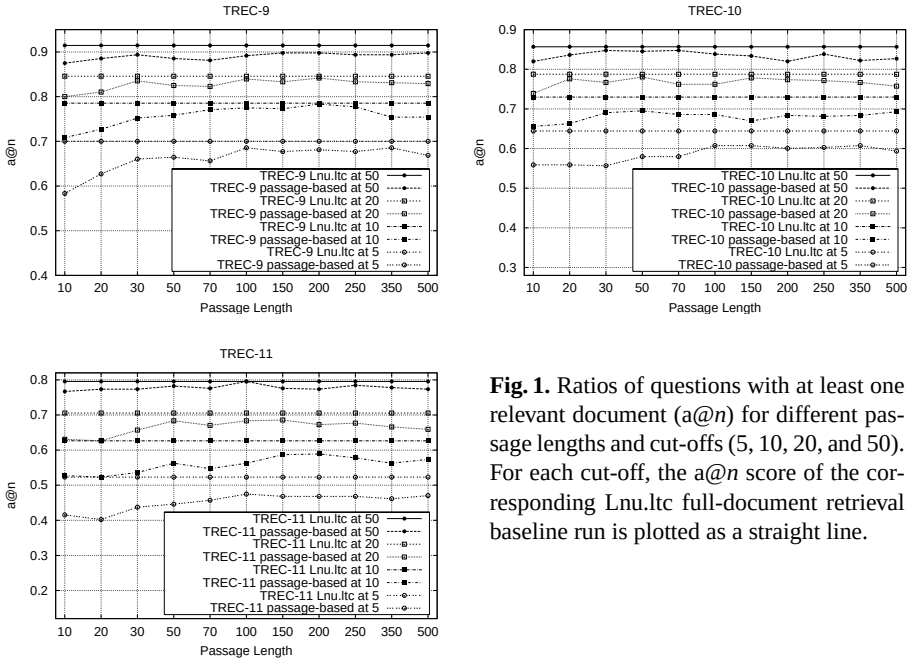

Fig. 1. Ratios of questions with at least one relevant document (a@n) for different passage lengths and cut-offs (5, 10, 20, and 50). For each cut-off, the a@n score of the corresponding Lnu.ltc full-document retrieval baseline run is plotted as a straight line.

Contrary to what one might expect, all runs using passage-based retrieval perform worse than the respective full-document retrieval run, at any cut-off. In none of the cases, passage-based retrieval provides more questions with at least one relevant document than full-document retrieval. We expected passage-based retrieval to improve early precision by preferring documents that contain matching terms closer to each other and rank lower documents that do contain terms in the query but the terms are more spread. To analyze whether precision increased, we measured the p@n score and some of the findings are shown in table 4. Unfortunately, due to space restriction, we can not display the results for all passage sizes, but we tried to select some window sizes that show the overall characteristics.

	p@*n*	full	Passage Length 30	70	150	250
TREC-9	p@5	0.3104	0.2721 (−12.33%)	0.2750 (−11.4%)	0.2767 (−10.85%)	0.2750 (−11.4%)
	p@10	0.2388	0.2085 (−12.68%)	0.2210 (−7.45%)	0.2196 (−8.03%)	0.2221 (−6.99%)
	p@20	0.1717	0.1617 (−5.82%)	0.1644 (−4.25%)	0.1637 (−4.65%)	0.1631 (−5.0%)
	p@50	0.1023	0.1021 (−0.19%)	0.1023 (±0.0%)	**0.1029 (+0.58%)**	0.1010 (−1.27%)
TREC-10	p@5	0.2707	0.2259 (−16.54%)	0.2411 (−10.93%)	0.2480 (−8.38%)	0.2494 (−7.86%)
	p@10	0.2127	0.1841 (−13.44%)	0.1885 (−11.37%)	0.1880 (−11.61%)	0.1892 (−11.04%)
	p@20	0.1542	0.1389 (−9.92%)	0.1386 (−10.11%)	0.1450 (−5.96%)	0.1417 (−8.1%)
	p@50	0.0886	0.0856 (−3.38%)	0.0851 (−3.94%)	0.0849 (−4.17%)	0.0843 (−4.85%)
TREC-11	p@5	0.1675	0.1415 (−15.52%)	0.1451 (−13.37%)	0.1508 (−9.96%)	0.1473 (−12.05%)
	p@10	0.1237	0.1068 (−13.66%)	0.1086 (−12.2%)	0.1147 (−7.27%)	0.1097 (−11.31%)
	p@20	0.0845	0.0802 (−5.08%)	0.0788 (−6.74%)	0.0791 (−6.38%)	0.0787 (−6.86%)
	p@50	0.0475	**0.0491 (+3.36%)**	**0.0487 (+2.52%)**	**0.0477 (+0.42%)**	0.0468 (−1.47%)

Table 4. p@*n* scores for different passages sizes compared to full-document retrieval.

Although precision does increase in a few cases, in general, also precision score drop when applying passage-based retrieval. Here, an increase in precision does not mean that more question are provided with relevant documents, as can be seen in figure 1, but that for some questions more relevant documents are found by passage-based retrieval than by full-document retrieval.

It is not obvious why passage-based retrieval performs worse than document retrieval. Especially since Llopis et al. [6] report significant improvements for passage-based retrieval when used for question answering: a@5 +11.26%, a@10 +14.28%, a@20 +13.75% and a@50 +9.34%. These improvements are with respect to the results of AT&T's version of SMART on the TREC-9 data set. It is hard to compare their results directly to ours for two reasons: First, the AT&T run is significantly worse than our baseline, and, secondly, it is not clear how they dealt with question variants, as discussed in section 2.1. Nevertheless their improvements are so large, that it is not unlikely that they also apply to our experimental setup.

In the approach by Llopis et al., documents are split into passages of n sentences ($n \in \{5, 10, 15, 20\}$), and each passage starts at the second sentence of the previous passage. Their improvements are probably not so much due to the fact that they use sentences instead of words to identify passage boundaries, but the fact that their passages have a much larger overlap ratio than the passages used here. Their best results are reported for passages containing 20 sentences, yielding an overlap ratio of approx. 95%—*approximately*, because sentences can differ in length—compared to an overlap of 50% used in our experiments.

Combining our results with the findings of Llopis et al., it can be concluded that passage-based retrieval can yield better results for document pre-fetching, but that passages should significantly overlap with each other.

4 Conclusions

In this paper, we evaluated the performance of three retrieval techniques with respect to question question answering: stemming, blind feedback, and passage-based retrieval. Applying stemming resulted in consistent improvements in precision and recall. Blind feedback performed rather badly, and should be discarded as an option for question answering.

Passage-based retrieval did not live up to the expected improvements. In fact, our approach resulted only in a few cases in minor improvements in precision, and overall performed worse than the baseline. This is in contrast to some other results in the literature and shows that the way passages are formed is an important issue.

Acknowledgments. The author was supported by the Physical Sciences Council with financial support from the Netherlands Organization for Scientific Research (NWO), project 612-13-001.

References

1. J. Callan. Passage-retrieval evidence in document retrieval. In B. Croft and C. van Rijsbergen, editors, *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 302–310, 1994.
2. C. Clarke, G. Cormack, D. Kisman, and T. Lynam. Question answering by passage selection (MultiText experiments for TREC-9). In E. Voorhees and D. Harman, editors, *Proceedings of the Ninth Text REtrieval Conference (TREC-9)*, pages 673–683. NIST Special Publication 500-249, 2000.
3. D. Hull. Stemming algorithms—a case study for detailed evaluation. *Journal of the American Society for Information Science*, 47(1):70–84, 1996.
4. M. Kaszkiel and J. Zobel. Effective ranking with arbitrary passages. *Journal of the American Society for Information Science and Technology*, 52(4):344–364, 2001.
5. W. Kraaij and R. Pohlmann. Viewing stemming as recall enhancement. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 40–48, 1996.
6. F. Llopis, A. Ferrández, and J. Vicedo. Passage selection to improve question answering. In *Proceedings of the COLING 2002 Workshop on Multilingual Summarization and Question Answering*, 2002.
7. C. Monz and M. de Rijke. The University of Amsterdam at CLEF 2001. In *Working Notes for the Cross Language Evaluation Forum Workshop (CLEF 2001)*, pages 165–169, 2001.
8. M. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.
9. Z39.50/Prise 2.0. `www.itl.nist.gov/iaui/894.02/works/papers/zp2/zp2.html`.
10. H. Schmid. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing*, 1994.
11. J. Xu and B. Croft. Query expansion using local and global document analysis. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 4–11, 1996.