Addressing Term Mismatch in Information Retrieval by Using Document Expansion

Nathan D. Beach
Computer Science Department
Calvin College
Grand Rapids, MI 49546, USA
ndbeach@gmail.com

Abstract

A common approach to solving the term mismatch between a user's query and relevant documents is query expansion, which augments a user's query with additional, related terms in order to increase coverage of relevant documents. The system proposed in this paper presents an alternative means of ameliorating the term mismatch problem for certain corpora. Instead of using query expansion, this system uses *document expansion*. Specifically, for corpora containing inter-document references (e.g., links or citations), this system uses those references to expand the set of terms contained within a given document. Preliminary evaluation of the results of using document expansion in a sample implementation indicates that this approach can provide mild benefit to the perceived helpfulness of a user's query.

1 Introduction

One fundamental problem in information retrieval is that there is often a mismatch between the terms of a user's query and the terms that compose relevant documents. This term mismatch may arise when a user isn't familiar with the appropriate terms or when some documents don't use the meta-terms that accurately characterize their content. For example, a Shakespeare play may contain a rising action, climax, or protagonist even if those meta-terms do not appear in the text itself.

Addressing this problem is important because new or inexperienced users are more likely to enter terms that do not match those used in the documents they are seeking (Furnas et. al., 1987) and because a document may be relevant to a particular concept even if the document does not use the corresponding meta-term. In both cases, resolving term mismatch is a key step to retrieving a comprehensive set of relevant documents.

The term mismatch problem is most often addressed using query expansion, a process that augments a query with terms that share a similar meaning so as to increase the likelihood of matching the terms contained in relevant documents. Query expansion—in its many different forms—is effective at addressing term mismatch caused by a user's unfamiliarity with the appropriate terms to use for a given concept (Cui et. al., 2002; Lima and Pedersen, 1999; Qiu and Frei, 1993; Voorhees, 1994; Xu and Croft, 1996; Yan and Hauprmann, 2007). However, the success of query expansion depends on its ability to augment a user's query with the correct terminology. For example, query expansion techniques that use a lexical ontology such as Wordnet (Miller, 1995) or a thesaurus are only as successful as the underlying data.

As an alternative to query expansion, the system described in this paper uses *document expansion*. This differs from query expansion in that it augments the terms contained within a document rather than a query. Specifically, document expansion capitalizes on the fact that many corpora have a significant number of inter-document references (e.g., links or citations). That is, there are many corpora containing some documents, which I call *primary documents*, that are linked to or referenced by other documents, which I label *second-*

ary documents. Document expansion uses those inter-document references to expand the terms contained within the primary documents in an effort to address the term mismatch problem.

In this paper, I will introduce the concept of document expansion in the context of related work, examine the types of corpora on which this system can be applied, and explain the means by which this model gleans information based on inter-document references. I will then present a preliminary evaluation of the system's performance and discuss possible future work.

2 Related Work

Document expansion addresses the same problem as query expansion—namely, the term mismatch problem. There are three common means of performing query expansion:

- *Global query expansion* uses corpus-wide statistics. Specifically, Voorhees (1994) uses a lexical ontology such as Wordnet to insert synonyms of a term. For example, a query containing *golf stroke* might be expanded to include terms such as *swing*, *slice*, *hook*, and *shot*. Qiu and Frei (1993) found benefits to expanding a query by adding terms similar to the concept of the query rather than adding synonyms of the terms in the query. However, if the term being searched for doesn't exist in the lexical ontology being used, then global query expansion provides no additional benefit.
- *Local query expansion* uses query-specific statistics. It assumes the top-ranked results are fairly relevant and then returns the results of a second, expanded query that uses terms garnered from the top-ranked results of the first query (Xu and Croft, 1996). Because this approach uses the results of the first search to generate terms used in the second search, local query expansion obviously provides no benefit when the user's search term(s) do not appear in any documents.
- *Query expansion using query logs* derives query expansion terms by analyzing the large quantities of user interaction data stored in a query log. This technique has proven effective for information retrieval on the Web, but obviously is only applicable to environments with significant user interaction data (Cui et. al., 2002; Lima and Pedersen, 1999).

By contrast, document expansion exploits the inter-document references in some corpora to augment the terms of a document. The Web is one example of a corpus with a significant number of inter-document references, and the World Wide Web Worm (WWWW) was one of the earliest search engines to use the text of a hyperlink to gain information about the page to which that hyperlink points (McBryan, 1994). More recently, Google, like most other modern Web search engines, associates the text of a hyperlink with the page to which that hyperlink points (Brin and Page, 1998).

The document expansion approach presented in this paper differs from the approach taken by WWWW and Google primarily in that both of those Web search engines use only the text of inter-document references but not the terms surrounding those references.[1] Additionally, document expansion is not limited solely to the Web; this system can be customized to recognize any type of reference.

3 Applicable Corpora

A corpus on which document expansion may be used must meet two fundamental criteria:

1. Some or all of the documents in the corpus must contain references to other documents within the corpus.
2. Those inter-document references must be formatted in some consistent manner such that they can be automatically recognized by a lexical parser.

I will give three specific examples of corpora that meet the aforementioned criteria.

- The Perseus Digital Library contains hundreds of Greco-Roman documents, some of which are primary and others secondary (2007). The primary documents in that library include well-known works such as Homer's *Illiad* and Virgil's *Aeneid* and lesser-known works such as Thucydides's *The Peloponnesian War*. The secondary documents include works such as Charles Smith's *Commentary on Thucydides*, which references *The Peloponneasian War* and other many works.

[1] According to Brin and Page (1998), Google only uses hyperlinked text—not the term surrounding the hyperlink—to gain information about the page to which a hyperlink points. As far as the author knows, this is still true.

- The Christian Classics Ethereal Library (CCEL, 2007) contains hundreds of texts—the secondary documents—that are marked with references to certain verses in the Bible—the primary documents.
- If one considers the set of all possible dates (from infinitely BCE to infinitely CE) to be primary documents, then the set of any news or encyclopedia articles could be used as secondary documents. This unique application of document expansion would permit one to search for a term such as "Martin Luther King birthday" and retrieve the primary document "15 January 1929."

All three of these corpora are suitable for document expansion because they contain consistently formatted inter-document references. Note that in all three of the aforementioned corpora the inter-document references are consistently formatted but need *not* be hyperlinked. Unlike Web-only search engines, document expansion enables the system to capitalize on non-hyperlinked references.

4 Model

For any single corpus, the set of primary documents is defined as the set of all documents *to which* there are inter-document references, and the set of secondary documents is defined as the set of all documents *in which* there are inter-document references. In most cases, corpora will contain documents that are members of both sets.

4.1 Building the Table of References

The first step in building the model is to create a table of references to store information about the inter-document references.

4.1.1 Algorithm

The procedure to construct the table of references is detailed in Algorithm 1.

Building the table of references requires sets containing all primary (P) and secondary documents (S) as well as the number of term buckets to create (N). The term bucket T_i is used to store the terms that have a distance of i terms before or after the inter-document reference. (See Figure 2 for an example table of references.)

Input: P, the set of all primary documents
Input: S, the set of all secondary documents
Input: N, the number of term buckets
Output: R, the table of references
1: Initialize R to contain the following columns: *PDName*, *PDContents*, *Title*, T_1, T_2, … T_N.
2: **for** p in P **do**
3: Add row for p to R
4: $R_{p,PDName} \leftarrow$ name of p
5: $R_{p,PDContents} \leftarrow$ contents of p
6: **end for**
7: **for** s in S **do**
8: Let s be viewed as an ordered list of M terms (words) such that s_i is the i^{th} token of s.
9: **for** each reference, r, to a primary document, p, in s **do**
10: **if** r occurs in the title of s **then**
11: Append $s_1, s_2, \ldots s_M$ to $R_{p,Title}$
12: **else**
13: $t \leftarrow$ position of r in s
14: **for** $i = 1$ to N **do**
15: Append s_{t-i} and s_{t+i} to $R_{p,Ti}$
16: **end for**
17: **end if**
18: **end for**
19: **end for**

Algorithm 1. Procedure to construct the table of references.

The table of references (R) contains exactly one row for each primary document and $N+3$ columns. The *PDName* column contains some identifier for the primary document, and the *PDContents* column contains the contents (or terms) of the primary document. The *Title* column contains the terms of those secondary documents whose title includes a reference to the given primary document. The remaining N columns contain terms extracted from the secondary documents that are subsequently used in the document expansion of a respective primary document.

To construct the table of references, the system iterates over each inter-document reference in the set of secondary documents and inserts the terms surrounding that reference into the term buckets of the primary document to which that reference points (cf. lines 7-19 of Algorithm 1).

Unless the good Shepherd shall place me on his shoulders and carry me back to the fold my steps will totter, and in the very effort of rising I shall find my feet give way. I am the prodigal son, Luke 15:11-32, who although I have squandered all the portion entrusted to me by my father, have not yet bowed the knee in submission to him; not yet have I commenced to put away from me the allurements of my former excesses.

Figure 1. An excerpt from a secondary document in the CCEL corpus ("To Theodosius and the Rest of the Anchorites" by Jerome). The inter-document reference to Luke 15 is demarcated with a box.

PDName	PDContents	Title	T_1	T_2	T_3	…	T_{19}	T_{20}
Luke 14	…	…	…	…	…	…	…	…
Luke 15	Now the tax collectors …[2]	*[Empty]*	son who	prodigal although	the I	…	and the	totter knee
Luke 16	…	…	…	…	…	…	…	…

Figure 2. An excerpt from the table of references after a reference to Luke 15 in the secondary document in Figure 1 has been processed. This figure demonstrates the effect that one iteration of the loop that begins on line 9 has on the table of references.

4.1.2 Example

Consider an example from the CCEL corpus in which the secondary documents include "To Theodosius and the Rest of the Anchorites" (see Figure 1), the primary documents include Luke chapter 15 from the Bible, and $N = 20$. (In other words, the text in Figure 1 $\in S$, and Luke 15 $\in P$.) The excerpt in Figure 1 includes an inter-document reference to the primary document Luke 15. Figure 2 demonstrates the state of the table of references after the reference to Luke 15 has been processed—i.e., after one iteration of the loop that begins on line 9.

Notice that the T_1 field in the table of references contains the word immediately preceding the reference to the primary document ("son") as well as the word immediately following it ("who"). Fields T_2 … T_{20} are populated similarly. In this case the *Title* field is empty; however, had a reference to the primary document Luke 15 occurred in the title of some secondary document then the full contents of that secondary document would be placed in the *Title* field for Luke 15 (cf. line 11 of Algorithm 1). Finally, note that additional references to Luke 15 in the set of secondary documents would result in more terms being appended to each of the columns for Luke 15.

4.1.3 Implications for Document Expansion

In summary, for each primary document, the table of references contains not only the contents of that primary document, but also excerpts from secondary documents that reference (or "link to") the primary document. In this way, the system can achieve document expansion on a primary document by using excerpts from secondary documents that link to that primary document.

4.2 Searching the Table of References

At query time, each row in the table of references can be treated as a document with multiple fields. The standard vector space model in information retrieval as described in Manning et. al. (2007) can then be used to search the primary documents. Note that, instead of searching only the *PDContents* column, we achieve document expansion by searching all the columns in the table of references.

4.2.1 Handling Document "Overexpansion"

Matching terms contained in any of the columns in the table of references can introduce matches that are "false positives" into the result set.

[2] This field contains the full text of the primary document. Note that it doesn't include the phrase "prodigal son," a common meta-term used to refer to this text.

One response to this concern is to rank the results using term frequency-inverse document frequency (*tf-idf*), as shown in equation (1). In general, this will return the most "relevant" documents first, pushing the "false positives" lower in the set of results returned.

$$score(t,d) = \sum_{f\ in\ F} tf(t,f) * idf(t) * boost(f) \quad (1)$$

where t = term, d = primary document, F is the set of all fields in the table of references for document d, and $boost(f)$ returns a boost value as explained in Section 4.2.2.

4.2.2 Field-Specific Boosts

The advantage to storing the terms surrounding an inter-document reference in separate fields T_1 … T_N is that it allows the system to place greater weight on terms closer to a reference (i.e., on the contents of T_1 and T_2) than on terms further from a reference (i.e., on the contents of T_{N-1} and T_N). Equation (2) defines a boost function that would accomplish such a weighting of the terms.

$$boost(f) = \begin{cases} 1 & \text{if } f \epsilon \{PDContents, Title\} \\ \frac{4}{f_p} & \text{otherwise} \end{cases} \quad (2)$$

where f = field and f_p = field proximity to inter-document reference ($1 \leq f_p \leq N$ and f_p is an integer).

5 Implementation

Since 2001, I have developed and maintained a Web site called Christ Notes,[3] which contains a Bible search in which the text of the Bible is stored in a MySQL database and is searched using SQL's LIKE clause. The results are then returned in their "natural" Biblical order (i.e., Genesis, Exodus, Leviticus, etc.).

I used Lucene[4] to create a Bible search that served as a sample implementation of document expansion. Specifically, I used six secondary documents from the CCEL corpus, and I used each Bible translation available on Christ Notes as a primary document.[5] Additionally, I used ten term buckets (N = 10) and Equation (2) as my boost function.

Lastly, I kept the MySQL-based search as the default Bible search and instead added the document expansion-based search as an alternative option, branding it as "concept match." Users may turn concept match on or off as desired.

6 Evaluation

In this section I present a preliminary evaluation of the aforementioned implementation of document expansion. The experimental setup is explained in Section 6.1, and the results are presented and discussed in Section 6.2.

6.1 Experimental Setup

To evaluate this implementation of document expansion, I conducted a discount usability study (Nielsen, 1993). I designed the experiment such that I split nine introductory computer science students into two groups: a control group ("Baseline"), in which concept match (i.e., document expansion) was turned off, and an experimental group ("Document Expansion"), in which concept match was on. Four students were in the Baseline group; five in Document Expansion. Students were not aware that different algorithms even existed let alone which one was being used to present results to them. The Bible search used by the two groups was identical in every respect except for the underlying search algorithm being used.

Each student was given five tasks. For each task the student was asked to pretend he was either researching a particular topic or preparing a sermon on that topic and to use the Bible search to accomplish that task.

I observed both quantitative and qualitative data. The former consisted primarily of a user's response to the question "How helpful do you think these results are for completing your task?" The responses were placed on a scale of 1 (very un-

[3] http://www.christnotes.org/
[4] http://lucene.apache.org/

[5] The secondary documents that I used were *Confessions of Saint Augustine*; *Easton's Bible Dictionary*; *Jamieson, Fausset, and Brown's Commentary on the Whole Bible*; *Nave's Topical Bible*; *Smith's Bible Dictionary*; and *Torrey's New Topical Textbook*. The Bible translations that I used as primary documents were the American Standard Version, Bible in Basic English, Darby's Translation, King James Version, World English Bible, and Young's Literal Translation.

helpful) to 5 (very helpful). The latter consisted of information such as a user's verbal feedback to me, a user's scrolling patterns, and user's search patterns.

6.2 Results

6.2.1 Quantitative Results

The quantitative results of this discount usability study are presented in Figure 3. Due to the small sample size of the study (nine students), the standard deviation of the helpfulness ratings is 0.85. Thus, the difference in perceived helpfulness between the Baseline and Document Expansion algorithms is not statistically significant.

Algorithm	Perceived Helpfulness
Baseline	3.58
Document Expansion	3.97

Figure 3. Responses of the users who evaluated the perceived helpfulness of search results. A rating of 1 indicates very unhelpful results; 5 indicates very helpful.

6.2.2 Qualitative Results

The discount usability study was very beneficial for gleaning comments and feedback from users. Below are three qualitative observations I made during the study:

- A user adapts to her expectations about a search engine. For example, when asked to research the Trinity, one user in the Baseline group did not even bother searching for *trinity* but instead immediately searched *father son spirit* knowing that the former would return no results. The very notion of concept match is that the search engine adapts to the expectations of a typical novice user.
- Three users complained that the number of results (often more than five hundred) was overwhelming. For users in the Document Expansion group, this complaint essentially amounts to an assertion of document "overexpansion." To address this, I subsequently extended the Lucene search engine to implement a "floor" such that the only documents returned are those whose relevancy scores for a particular search exceed the floor.
- Users really appreciated that the terms for which they searched were highlighted on the search results page. However, five users suggested that I also highlight those same terms when a user views the full contents of a chapter of the Bible after having clicked on a link to that chapter from the search results page. Although I already knew that this feature would be a nice addition, the fact that more than half of my testers voluntarily suggested this very same idea prompted me to make this full-chapter highlighting one of the very first features I subsequently implemented.

7 Conclusion

A common approach to addressing the term mismatch problem is to use query expansion. I presented an alternative approach—document expansion. This approach uses the references that exist in some documents to other documents to expand the set of terms for which a document matches. I then discussed a fully-functional implementation of document expansion on the Web site Christ Notes. A preliminary discount usability study of this implementation indicated that document expansion, as implemented on Christ Notes, provided a mild—although not statistically significant—improvement of the helpfulness of results returned. For this reason, it seems prudent to release this document expansion system to the public.

8 Future Work

The results of the preliminary evaluation warrant a more thorough study of document expansion. If the results of a comprehensive evaluation are positive, then there are several areas in which future work would be appropriate:

- When implementing this system, I used the boost function given in Equation (2). However, the choice of that specific boost function was based only on "eyeballing" the perceived relevance of the results—not on a systematic, objective measurement. Further work should be done to investigate an optimal boost function.
- One weakness of this approach to document expansion is its tendency to produce many "false positive" matches. Although using *tf-idf* to rank results is a good first step toward addressing this problem, further work should be

done to examine additional means of countering document "overexpansion." For example, one could investigate ways to include only certain terms when expanding a document perhaps through an analysis of term co-occurrence in the secondary documents.

- This paper deals exclusively with document expansion. Further work should be done to investigate effective means of combining document expansion with query expansion. For example, one could study the potential benefits or drawbacks of combining document expansion with techniques such as local context analysis (Xu and Croft, 2000) or probabilistic local feedback (Yan and Hauprmann, 2007).
- As presently implemented, a typical search using concept match on Christ Notes requires about 1.5 seconds.[6] Such a delay is tolerable, but it is far from ideal. Although concept match is fully implemented on the Christ Notes test server, I have yet to deploy this implementation to the production server so that I can research ways to decrease the average length of time required for a search using concept match. Once I satisfactorily decrease search latency, I intend to deploy concept match to the Christ Notes production server.

Acknowledgments

The author is grateful to Keith Vander Linden for advising him, Matt Johnson for his invaluable suggestions, and the nine introductory computer science students who participated in the discount usability study for their helpful feedback.

References

Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual Web search engine. *Proceedings of the seventh international conference on World Wide Web*, Brisbane, Australia.

Christian Classics Ethereal Library. 2007. Calvin College, Grand Rapids, MI, USA. http://www.ccel.org/.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schutze. 2007. *An Introduction to Information Retrieval*. Cambridge University Press, Cambridge, UK.

Hang Cui, Ji-Rong Wen, Jian-Yun Nie, and Wei-Ying Ma. 2002. Probabilistic query expansion using query logs. *Proceedings of the 11th international conference on World Wide Web*, Honolulu, Hawaii, USA.

George W. Furnas, Thomas K. Landauer, Louis M. Gomez, and Susan T. Dumais. 1987. The vocabulary problem in human-system communication. *Communications of the ACM*, 30(11):964-971.

Erika F. de Lima and Jan O. Pedersen. 1999. Phrase recognition and expansion for short, precision-biased queries based on a query log. *Proceedings of SIGIR'99*, Berkeley, California, USA.

Oliver A. McBryan. 1994. GENVL and WWWW: Tools for Taming the Web. *Proceedings of the first International World Wide Web Conference*, Geneva, Switzerland.

George A. Miller. 1995. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39-41.

Jakob Nielsen. 1993. *Usability Engineering*. Academic Press. San Diego, CA.

Perseus Digital Library. 2007. Tufts University, Boston, MA. http://www.perseus.tufts.edu/.

Yonggang Qiu and Hans-Peter Frei. 1993. Concept based query expansion. *Proceedings of SIGIR'93*, Pittsburgh, PA.

Ellen M. Voorhees. 1994. Query expansion using lexical-semantic relations. *Proceedings of SIGIR'94*, Dublin, Ireland.

Jinxi Xu and W. Bruce Croft. 1996. Query expansion using local and global document analysis. *Proceedings of SIGIR'96*, Zurich, Switzerland.

Jinxi Xu and W. Bruce Croft. 2000. Improving the effectiveness of information retrieval with local context analysis. *ACM Transactions on Information Systems*, New York, New York, USA.

Rong Yan and Alexander Hauprmann. 2007. Query expansion using probabilistic local feedback with application to multimedia retrieval. *Proceedings of CIKM'07*, Lisbon, Portugal.

[6] This is measured as the length of time from when the HTTP request is received by the server to when the complete body of the HTTP response has been sent back to the client.