Automatic Term Reweighting for Query Expansion

Reuben Crimp
University of Otago
Dunedin, New Zealand
rcrimp@cs.otago.ac.nz

Andrew Trotman
University of Otago
Dunedin, New Zealand
andrew@cs.otago.ac.nz

ABSTRACT

Query expansion is used to overcome the vocabulary mismatch between the documents and queries, but it can lead to query drift. We propose an automatic term reweighting strategy for BM25 ranking functions. Using expansion terms obtained from general purpose thesauri, we found that reweighting through *term frequency merging* is more effective than standard query expansion. Instead of appending the new terms directly to the original query, we merge the term frequencies with the original query term. This reduces the impact of spurious expansion terms being over represented in the modified query.

CCS CONCEPTS

•Information systems → Query reformulation;

KEYWORDS

Ad-hoc Retrieval, Query Expansion, tf-merging, Thesaurus, WordNet, Roget

ACM Reference format:
Reuben Crimp and Andrew Trotman. 2017. Automatic Term Reweighting for Query Expansion. In *Proceedings of The 22nd Australasian Document Computing Symposium, Brisbane, QLD, Australia, December 7–8, 2017 (ADCS 2017),* 4 pages.
DOI: 10.1145/3166072.3166074

1 INTRODUCTION

The vocabulary mismatch problem is rampant in human generated text, even domain experts will use different terms to describe the same concept. Experiments have show that the likelihood of two such individuals using the same term is less than 20% [3]. In information retrieval research this problem is often described as *term mismatch*, because there is a disparity between the terms used in search queries and the terms used in a document corpus. This is a problem when you consider that the average Internet search query is reported to be less than 4 terms. With fewer words the probability of terms coocurring within documents is clearly lower than a more comprehensive search query. Analysis has also shown that even when experts manually tag queries to relevant documents, many query terms will not be contained within those documents. This has been shown in the TREC ad hoc retrieval tracks [15].

Reformulating a search query is a typical technique used to address term mismatch, the most common of which is *query expansion*. Additional terms are appended to the search query, terms which are both relevant to the query and frequent in relevant documents. The combination of the original query terms and the expanded terms should more accurately reflect the vocabulary used in the corpus. A popular query expansion algorithm is Rocchio's *relevance feedback* technique [9]. The unmodified query is used to retrieve a short list of relevant documents, then terms that strongly represent those documents (discriminating terms) are extracted and ranked in descending order of weight. The top terms are then appended to the query, and the search process is run a second time. This method is known as *blind relevance feedback* as the process does not require a user to manually identify the relevant discriminating terms. Because a document can cover many different topics, even the most relevant documents can include terms which are irrelevant to the users information need. Appending these spurious terms can cause *query drift*, which leads to performance degradation. Refining the initial retrieved document set can help reduce the effect of drift [8]. Accounting for the terms locality within the document with respect to the original query terms can also help alleviate this issue [14]

Using a precomputed thesaurus is another way to obtain expansion terms, and also faster than relevance feedback which requires multiple passes through the search engine pipeline. Domain specific thesauri are prohibitively expensive to construct manually, since they require experts with domain knowledge of the documents to label the data. However, some exist such as the Unified Medical Language System (UMLS), which contains terms from 40 biomedical vocabularies [4]. Using general-purpose thesauri (like Roget's or WordNet) for query expansion was tested in the early days of information retrieval. Early evidence suggested retrieval accuracy could be improved , but that it was too unpredictable to be useful in practice [10]. Voorhees concluded that lexical-semantic relationships provide little benefit, but have the 'potential to improve an initial query' [13].

No matter which method is used, automatic query expansion is not perfect and will often append terms that cause query drift. Blind relevance feedback can potentially identify terms that are in relevant documents but not relevant to the original query. A thesaurus can identify terms synonymous to alternatives definitions. e.g. the polyseme 'mine', could refer to explosives or an excavation site. This is not something which can be easily avoided in automatic systems, but we can change the way our ranking function treats expansion terms. Many ranking functions treat expansion terms with the same level of importance as the original user generated query terms, despite the fact that they have a higher chance of being spurious. In our experiments we compare standard query expansion to *term frequency merging*, and found it to be effective at combating the effects of query drift.

2 EXPERIMENT CONDITIONS

The purpose of this investigation was to determine if the effects of query drift could be reduced with term reweighting. We used the TREC ad-hoc retrieval tracks 1 to 8. Approximately 1,367,000 documents (excluding CR) from TREC disks 1 to 5, queries 51 to 450, and the set of binary relevance evaluations. The ATIRE search engine [11] was used in our experiments.

2.1 General-Purpose Thesauri

Two general-purpose thesauri were used in our experiments. Rogets thesaurus which only identifies synonyms; we used the 2004 Project Gutenburg edition, which contains over 36,000 words.

The other was WordNet, one of the more comprehensive general-purpose thesauri available today. Constructed at Princeton by Miller and his team in 1985 [7]. The relationships in WordNet are more precisely catagorised than Roget's, it has 26 different relationship types. Some of these relationships are symmetric, where the associated terms are in a *synset*, or *synonym ring*. Other relationships are not, and describe an 'is-a' or 'has-a' relationship.

Many IR researchers have used older versions of this thesaurus for query expansion, including Voorhees who used version 1.3 [13]. We used WordNet 3.1 the most recent version available.

2.2 Ranking Function

We used a variant of the BM25 ranking function, with parameters k_1 and b chosen empirically by particle swarm optimization [12].

$$RSV(d) = \sum_{t \in Q} log\left(\frac{N}{df_t}\right) \times \frac{(k_1 + 1) \cdot tf_{td}}{k_1 \cdot \left(1 - b + b \cdot \left(\frac{L_d}{L_{avg}}\right)\right) + tf_{td}} \qquad (1)$$

BM25 computes the retrieval status value (RSV) of the document by summing the contribution of each query term individually. Each term's contribution can be expressed as a product of two values, the inverse document frequency (IDF) component and the term frequency (TF) component.

We used the Robertson-Walker IDF component, $log(N/df_t)$, which prevents negative scores [5]. The IDF component scales the RSV by taking into account the frequency of a query term across the entire document collection. If a term is contained within the majority of the collection (e.g. 'the'), it is not considered a discriminating term, so its contribution to the RSV is negligible. But if the term is only contained within few documents, then it is considered highly discriminating, and has a large contribution to the final RSV.

The TF component is an asymptotic function which scales the contribution of the term frequency (tf_{td}). The contribution is also scaled by the length of the documents (L_d), relative to the average document (L_{avg}). The main purpose of this function is to prevent over boosting, by reducing the contribution of highly frequent terms. If this is not done we risk one term dominating the entire rank, and the less frequent terms having a relatively inconsequential contribution. Figure 1 shows the contribution of the TF component, with many of the parameters fixed for the sake of clarity. It is clear that the first few occurrences of term t have the largest effect, after $tf_{td} > 10$ the functions output has begun to plateau.

2.3 Query Expansion Modes

As described in Section 1, there are many ways to derive a set of expansion terms from a query. Knowing which documents contain these expansion terms allows more documents to be retrieved (improving recall), and also to compute a more accurate rank (improving average precision). We tested two separate methods of query expansion, the standard approach and tf-merging.

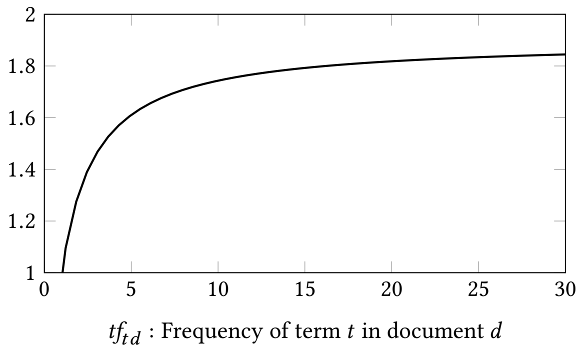
Standard query expansion reformulates the original query by appending the expansion terms as additional terms. This increases the diversity of the vocabulary and can potentially improve retrieval performance. Each expansion terms is treated just like one of the original query terms. The df_t and tf_{td} are calculated for the expansion term, and the contribution is added to the final RSV sum. The standard approach is simple but the query can easily drift towards one of the more frequent terms in the expansion, or become biased towards a subset of the original query.

If we define an *expansion-set* to be the set of expansion terms for a single term. e.g. for TREC-6 query 40 'land mine ban', we get three expansion-sets, one for each term in the unmodified query. Some expansion-sets might contain hundreds of discriminating terms, while other expansion-sets might contain only a few, or possibly none. It is possible that a large expansion-set could over represent the original term in the modified query, and the sum of the contributions could dominate the rank over smaller expansion-sets.

It would be appropriate to include the expansion terms within the TF component of BM25. This way we could take advantage of the diminishing returns seen in Figure 1. This way the existence of expansion terms in a document can increase the final RSV, without biasing the rank towards terms with excessive expansions.

We can achieve this with tf-merging, which is an automatic term reweighting approach. Instead of appending the new terms directly to the original query, the frequencies of the expansion terms are combined with the frequencies of the original terms. The IDF is consequently calculated more accurately for each individual query term because the terms postings lists are expanded rather than treating each expansion as a unique query term. Both tf_{td} and df_t become a sum/union over their respective expansion sets. A more precise explanation is provided in Section 3. From this point onwards tf-merging will refer to term reweighting by merging term frequencies, and query expansion will refer to the standard approach.

Figure 1: Example Output of the BM25 TF Component
$L_d = L_{avg}$, $k_1 = 0.9$, $b = 0.4$, asymptote at $k_1 + 1$.



3 TF-MERGING DETAILS

In (2) we define e_t to be the set of expansion terms of t. Where $t \mapsto u$ means u is an expansion term of t. It is possible that $t \not\mapsto t$ (e.g. hyponyms), so we explicitly include t.

$$e_t = \{u \mid t \mapsto u\} \cup \{t\} \tag{2}$$

For Roget's thesaurus, e_t is simply the synonyms of t.

3.1 Standard Query Expansion

Standard query expansion with BM25 (1) replaces the original query set Q, with the set of all expansion terms, shown in (3).

$$Q \to \bigcup_{t \in Q} e_t \tag{3}$$

3.2 tf-merging

For tf-merging we leave the query set as Q, but we redefine tf_{td} and df_t. tf_{td} becomes equation (4); the sum of term frequencies for each expansion. In other words, the number of times t or an expansion of t, occurs in document d.

$$tf_{td} \to \sum_{w \in e_t} tf_{wd} \tag{4}$$

df_t becomes equation (5); the magnitude of the set of documents ($d \in D$) which contains an expansion of t, or t itself. In other words, the number documents which contain t or an expansion of t.

$$df_t \to \left| \bigcup_{w \in e_t} \{d \mid w \in d\} \right| \tag{5}$$

Here is the full BM25 formula with tf-merging.

$$\sum_{t \in Q} log\left(\frac{N}{\left| \bigcup_{w \in e_t} \{d \mid w \in d\} \right|} \right) \times \frac{(k_1 + 1) \cdot \sum_{w \in e_t} tf_{wd}}{k_1 \cdot \left(1 - b + b \cdot \left(\frac{L_d}{L_{avg}}\right)\right) + \sum_{w \in e_t} tf_{wd}} \tag{6}$$

4 RESULTS

Table 3 shows the MAP of each TREC track using different query expansion techniques. The bold entries indicate an improvement over doing nothing. Rocchio's relevance feedback method shows that standard query expansion can consistently improve the mean average precision (MAP). However when using a general-purpose thesaurus (Roget/WordNet), standard query expansion degrades the MAP in all cases. These results are not unexpected. We can see that our tf-merging technique performs better than the standard approach, and occasionally performs better than doing nothing.

Focusing specifically on the thesaurus approach, standard query expansion improves about 57% of the 400 queries, and degrades 42%. Tf-merging however improves 71% of the queries and degrades only 29%. See Table 1 for the probability of improvement/degradation using specific WordNet relationships. By improved we mean no worse (i.e. same or better).

We performed two-tailed t-tests on 400 paired MAP samples. We tested the baseline against, standard query expansion, and tf-merging, and in every case we obtained p-values < 0.003. Which suggests that the observed differences cannot be attributed to chance alone.

Table 1: Probability of Improvement vs Degradation

	Query Expansion		tf-merging	
Relation	improved	dedgraded	improved	degraded
Antonym	0.5025	0.4975	0.6525	0.3475
Entailment	0.8800	0.1200	0.9325	0.0675
Hypernym	0.1175	0.8825	0.2450	0.7550
Hyponym	0.1875	0.8125	0.3975	0.6025
Meronym part	0.8925	0.1075	0.9050	0.0950
Meronym subs	0.7900	0.2100	0.9200	0.0800
Similar To	0.4650	0.5350	0.6200	0.3800
Roget	0.5875	0.4125	0.7250	0.2750
WordNet (all)	0.5768	0.4232	0.7095	0.2905

4.1 Evidence of Query Drift

By inspecting the expansion terms of individual queries, we can see exactly how query drift has been addressed. Table 2 shows 9 of the 170 hyponym expansions for the TREC-6 query 40 'land mine ban'. Some expansion terms like *c, e, f, g, i* are clearly relevant, so boosting documents containing these terms will improve our overall MAP. Other terms like *a, b, d* are clearly irrelevant. More importantly there are 153 expansions for the term 'land', many of which are geographic labels. The excessive amount of expansion terms compared with 'mine' and 'ban', caused the query to drift towards documents only about 'land'. However with tf-merging, the effects of query drift were greatly reduced, as the other two terms were not overshadowed by the 'land' term.

Without query reformulation the MAP is 0.0805, query expansion degrades this to 0.0014, and tf-merging improves the MAP to 0.0984. Standard query expansion clearly has a huge negative impact for this query, and tf-merging has a small positive effect.

It is likely that tf-merging will fail to reduce the effects of query drift, when there are many discriminating spurious terms in every expansion set. But if the expansion term selection process is good, then tf-merging should be able to improve it.

Table 2: Expansion hyponyms, TREC-6 query 40

Term	num	Example expansion terms		
land	153	crash land$_a$	farmland$_b$	no man's land$_c$
mine	18	goldmine$_d$	booby trap$_e$	ground emplacedmine$_f$
ban	9	embargo$_g$	rusticate$_h$	cease and desist$_i$

4.2 Speed

We did not focus our experiments on time efficiency but it was a consideration. Automatic query expansion methods like blind relevance feedback are intrinsically slow, as they require at least two passes through the search engine pipeline. The problem is worse in a distributed environment, as the document list from the first pass must be merged over a network before the second pass can be executed [6].

We found Rocchio to be much slower, than thesaurus based expansion. Tf-merging takes longer than standard query expansion. The mean time for Rocchio was 324 seconds, tf-merging was 122 seconds and standard query expansion was 102 seconds.

Table 3: Mean Average Precision

Expansion terms	mode	TREC-1	TREC-2	TREC-3	TREC-4	TREC-5	TREC-6	TREC-7	TREC-8
None (baseline)		0.2181	0.1993	0.2324	0.1727	0.1432	0.1891	0.1905	0.2195
Rocchio	query exp'	**0.2311**	**0.2103**	**0.2476**	**0.1802**	**0.1468**	**0.1925**	**0.2007**	**0.2286**
Antonym	query exp'	0.1977	0.1823	0.2139	0.1411	0.1230	0.1618	0.1741	0.1977
Entailment	query exp'	0.2153	0.1942	0.2315	0.1651	0.1311	0.1851	0.1898	0.2171
Hypernym	query exp'	0.1220	0.0680	0.1139	0.0335	0.0475	0.1058	0.1011	0.1155
Hyponym	query exp'	0.1258	0.1114	0.1161	0.0243	0.0362	0.1347	0.1068	0.0937
Meronym part	query exp'	0.2154	0.1972	0.2250	0.1642	0.1402	0.1886	0.1896	0.2076
Meronym subs	query exp'	0.2157	0.1839	0.2168	0.1593	0.1259	0.1860	0.1911	0.2095
Similar To	query exp'	0.1760	0.1630	0.1754	0.1065	0.1074	0.1714	0.1715	0.2084
Roget	query exp'	0.1995	0.1740	0.1945	0.1349	0.1119	0.1802	0.1853	0.2069
Antonym	tf-merging	0.2157	**0.2004**	0.2292	0.1718	0.1404	0.1851	0.1895	0.2161
Entailment	tf-merging	0.2180	0.1978	0.2324	0.1726	0.1319	0.1879	0.1900	0.2175
Hypernym	tf-merging	0.1446	0.1414	0.1553	0.1072	0.1012	0.1205	0.1104	0.1716
Hyponym	tf-merging	0.1940	0.1929	0.1846	0.1483	0.1292	0.1844	0.1686	0.1996
Meronym part	tf-merging	**0.2190**	**0.2022**	0.2307	0.1694	0.1406	**0.1912**	**0.1912**	0.2152
Meronym subs	tf-merging	0.2180	0.1993	0.2324	0.1727	0.1425	0.1891	0.1834	0.2195
Similar To	tf-merging	0.2075	0.1959	0.2183	0.1659	0.1301	**0.1910**	0.1882	0.2041
Roget	tf-merging	0.2173	0.1986	0.2225	0.1609	0.1393	0.1877	0.1887	0.2041

5 CONCLUSIONS

The purpose of our experiments was to determine if using general-purpose thesauri for query expansion could be improved. We tested two different thesauri (Roget and Wordnet), with two different query expansion techniques. We compared standard query expansion with our proposed method that we call *tf-merging*.

Our results agree with previous experiments, using a general-purpose thesauri appears to have potential, but the usefulness is not immediate. Despite the fact that WordNet has become more comprehensive as a thesaurus, it is still not viable for basic query expansion. It *can*, and frequently does, improve the precision of ad-hoc retrieval tasks, but there is still a significant chance that the precision will be degraded.

Reformulating a query through tf-merging makes sense intuitively, as we are not biasing our search towards query terms which have more expansions. Modern improvements to query expansion use context, or word sense disambiguation to discern between good and bad expansion terms [2]. What we have proposed is straightforward and efficient, and it does not require a sophisticated language model to be implemented. Our approach of tf-merging does empirically improve general-purpose thesauri query expansion, but not enough to be competitive against other query reformulation methods.

Since our results indicate an improvement to thesaurus based expansion, in future work we will apply tf-merging to other query reformulation methods. The most obvious choice would be relevance feedback approaches, as their effectiveness is well established, especially the RM3 relevance model [1]. Since tf-merging is independent of the expansion term selection process, it could easily be used to complement existing query expansion improvements, like word sense disambiguation. We will also test the effectiveness with other ranking functions that use TF and IDF scores.

6 ACKNOWLEDGEMENTS

We thank the ADCS for providing an international travel grant.

REFERENCES

[1] N. Abdul-Jaleel, J. Allan, W.B. Croft, F. Diaz, L. Larkey, X. Li, M.D. Smucker, and C. Wade. 2004. UMass at TREC 2004: Novelty and HARD. In *TREC 2004*.

[2] J. Bai, D. Song, P. Bruza, J.-Y. Nie, and G. Cao. 2005. Query Expansion Using Term Relationships in Language Models for Information Retrieval. In *CIKM '05*. 688–695.

[3] G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. 1987. The Vocabulary Problem in Human-system Communication. *CACM* 30, 11 (1987), 964–971.

[4] B. L. Humphreys, D. A. Lindberg, H. M. Schoolman, and G. O. Barnett. 1998. The Unified Medical Language System: an informatics research collaboration. *J Am Med Inform Assoc* 5, 1 (1998), 1–11.

[5] L. Lee. 2007. IDF Revisited: A Simple New Derivation Within the Robertson-Spärck Jones Probabilistic Model. In *SIGIR '07*. 751–752.

[6] F. Martínez-Santiago, M. A. García-Cumbreras, and L. A. Ureña Lòpez. 2006. Does Pseudo-relevance Feedback Improve Distributed Information Retrieval Systems? *IP&M* 42, 5 (2006), 1151–1162.

[7] G. A. Miller. 1995. WordNet: A Lexical Database for English. *CACM* 38, 11 (1995), 39–41.

[8] M. Mitra, A. Singhal, and C. Buckley. 1998. Improving Automatic Query Expansion. In *SIGIR '98*. 206–214.

[9] J. J. Rocchio. 1971. Relevance feedback in information retrieval. In *The Smart retrieval system - experiments in automatic document processing*, G. Salton (Ed.). Englewood Cliffs, NJ: Prentice-Hall, 313–323.

[10] G. Salton and M. E. Lesk. 1968. Computer Evaluation of Indexing and Text Processing. *J. ACM* 15, 1 (1968), 8–36.

[11] A. Trotman, C. L. A. Clarke, I. Ounis, S. Culpepper, M.-A. Cartright, and S. Geva. 2012. Open Source Information Retrieval: A Report on the SIGIR 2012 Workshop. *SIGIR Forum* 46, 2 (2012), 95–101.

[12] A. Trotman, A. Puurula, and B. Burgess. 2014. Improvements to BM25 and Language Models Examined. In *ADCS '14*. 58:58–58:65.

[13] E. M. Voorhees. 1994. Query Expansion Using Lexical-semantic Relations. In *SIGIR '94*. 61–69.

[14] J. Xu and W. B. Croft. 2000. Improving the Effectiveness of Information Retrieval with Local Context Analysis. *TOIS* 18, 1 (2000), 79–112.

[15] L. Zhao and J. Callan. 2010. Term Necessity Prediction. In *CIKM 2010*. 259–268.