Preliminary Research Report
Metasearch Engine

Author: 12254530

21^{st} June 2013

1 Aggregation Technique

The main aggregation technique that I propose to use to combine the results returned from each retrieval system is Reciprocal Rank Fusion (RRF).

1.1 Description of Technique

Reciprocal Rank Fusion (RRF) merges documents returned into a unified list using only the rank positions of the retrieved documents. Theses rank positions are essentially determined by the search engines themselves, where a simple score for each document is calculated based on it's position in each result set. When a duplicate document is found, the inverse of it's rankings are summed up, since documents returned by more than one retrieval system might be more likely to be relevant. Systems that are not ranking a document are skipped [1].

1.2 Motivation

On advantages of using the RRF method is that it does not require relevance scores, unlike combSUM and combMNZ. In many real-world situations, relevance scores are not available [2]. As well as this, RRF requires no special voting algorithm or global information [3]. Unlike Borda Fusion which uses positional algorithms, and Condorcet Fusion which uses majoritarian algorithms [4].

Ranks may be computed and summed one system at a time, avoiding the necessity of keeping all rankings in memory. According to Cormack [3], RRF consistently yields better results than any individual system, and better results than the standard method Condorcet Fuse. It is conjectured that RRF outperforms Condorcet because it is better able to harness diversity within individual rankings. With Condorcet, a simple majority of weak preferences may overrule substantially stronger ones [3].

1.3 Implementation Outline

To implement the Reciprocal Rank Fusion technique, one computes the rank score r of say document d_i, using the following formula:

$$r(d_i) = \frac{1}{\sum_j 1/k(d_{ij})} \tag{1}$$

where k = position information of d_i in all the systems $j \in \mathbb{N}$.

For instance, say three different search engines, A, B, and C, with a document collection compose of a, b, c, d, e, f, and g, return the following set of results for a particular search.

$A = (a, b, c, d)$
$B = (a, b, c, f, g)$
$C = (c, a, f, e, b, d)$

To compute the rank score of document a, equation 2.1 would be implemented as follows:

$$\begin{aligned} r(a) &= \frac{1}{1/k(d_{aA}) + 1/k(d_{aB}) + 1/k(d_{aC})} && (2)\\ &= \frac{1}{(1/1) + (1/1) + (1/2)} && (3)\\ &= \frac{1}{5/2} && (4)\\ &= \frac{2}{5} && (5)\\ &= 0.4 && (6) \end{aligned}$$

Therefore, the rank scores of the documents are as follows:

$r(a) = 0.40$
$r(b) = 1/(1/2) + (1/2) + (1/5) = 0.83$
$r(c) = 1/(1/3) + (1/3) + (1/1) = 0.60$
$r(d) = 1/(1/4) + (0) + (1/6) = 2.40$
$r(e) = 1/(0) + (0) + (1/4) = 4.00$
$r(f) = 1/(0) + (1/4) + (1/3) = 1.71$
$r(g) = 1/(0) + (1/5) + (0) = 5.00$

Thus, $a > c > b > f > d > e > g$.

Note the presence of 0 for d, e, f, and g. As systems not ranking a document are skipped [1].

The method itself is relatively simple. However, one difficultly which is likely to be encountered is the occurrence of a number of documents with the same rank scores. In order to overcome this, one may either display results as tied. Or one may need to abandon the assumption that all search engines are equal and apply a weighting to each, reflecting a hierarchy of the systems such as $A > B > B$, assuming A is the best performing and so forth.

2 Review of Query Preprocessing in Search Engines

Web search engines typically perform some kind of preprocessing on the queries they receive. The first step is to determine the encoding of a document and to filter out unwanted characters and mark-up, for instance, HTML tags, punctuation, numbers, etc. Then the text is broken into tokens (keywords) by using delimiters such as white space and punctuation characters. Tokens can be used as they are, or they can be transformed into a base form, for example nouns in the singular form, verbs in the infinitive form, etc. (e.g., books becomes book, talked becomes talk). A common approach is to stem the tokens. For example, computational becomes comput and computing becomes comput [5].

Tokens can also be normalised through synonym matching, where words with similar or the same meanings are grouped together such as fast, quick, speedy. Synonym lists can be quite labour intensive, generally being built by hand, but can give more control [6]. Another preprocessing step that is sometimes carried out, is to remove frequent words that appear in most of the documents and that do not bear any meaningful content. These are called stopwords, examples include 'a', 'the', 'it'. Finally, a search engine may also perform query expansion. Query expansion involves evaluating input text and expanding the search query to match additional documents.

2.1 Advantages of Preprocessing Techniques

These techniques are generally carried out to improve the quality and speed of processing. For instance, stemming the terms before building an inverted index has the advantage that it reduces the size of the index, and allows for retrieval of webpages with various inflected forms of a word. For example, when searching for webpages with the word computation, the results will include webpages with computations and computing. Stemming is easier

to do than computing base forms, because stemmers remove suffixes, without needing a full dictionary of words in a language [5].

2.2 Drawbacks of Preprocessing Techniques

However, there can be a trade-off for using certain preprocessing techniques. Stemming input terms leads to more documents being matched, increasing the total recall. Thus, reducing the precision of the returned information. This is also the case when a query is expanded to search for all the synonyms of input text, resulting in increased recall is at the expense of precision. Another problem that can be encountered, is when stopword lists are incorporated. Though they usually can be discarded safely, there are times when this is not always the case. For example, the phrase "to be or not to be" where every word is potentially a stopword. A system will potential return no results for this kind of query [6].

Author: Grayson-54530 // Email: 12254530@ucdconnect.ie

References

[1] Nuray, R., & Can, F. (2006). *Automatic ranking of information retrieval systems using data fusion. Information processing & management, 42*(3), 595-614.

[2] Aslam, J. A., & Montague, M. (2001, September). Models for metasearch. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 276-284). ACM.

[3] Cormack, G. V., Clarke, C. L., & Buettcher, S. (2009, July). Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval* (pp. 758-759). ACM.

[4] Montague, M., & Aslam, J. A. (2002, November). Condorcet fusion for improved retrieval. In *Proceedings of the eleventh international conference on Information and knowledge management* (pp. 538-548). ACM.

[5] Inkpen, D. (2007, December 8). Information Retrieval on the Internet. Retrieved from http://www. si-te. uottawa. ca/-diana/csi4107 L, 1.

[6] Skorkin, A. (2010, March 1). How Search Engines Process Documents Before Indexing. *A description into the preprocessing techniques performed by search engines.* Retrieved from http://www.skorks.com/2010/03/how-search-engines-process-documents-before-indexing/