Information Retrieval Practical

Joseph Root (MSc)

Introduction

This project looked to assemble a basic query system for effectively retrieving documents from a corpora of news articles. We will look at the effectiveness of the *cosine* method when retrieving documents, along with introducing the *Rocchio* algorithm for incorporating user feedback. Furthermore, we will experiment with other potential improvements and modifications such as down-casing and stemming.

Methodologies

Within practical 1 we implemented a basic *cosine* algorithm for scoring document-query similarities. Broadly speaking the query engine itself can be outlined as such:

1. Read and parse the index file in order to build our initial index for calculating cosine scores. Each term from the index is down-cased and stemmed, and the lengths are recalculated within code.
2. Read and parse the relevant document file, storing each of the relevant document IDs within an array.
3. When a query, *q*, is entered:
   a. *q* is down-cased before being split into its constituent terms. Each term is then stemmed.
   b. For each term, *t*, within *q*, we iterate through every document, *d*, the term exists within.
   c. The score for *d* is incremented with the *TF-IDF* score given *t*'s frequency within *d* and the corpora at large.
   d. All relevant documents are ordered and returned with their score.
4. In order to calculate the correct precision at recalls, we count the number of feedback documents, before calculating the required number of correct documents for each recall. The array of results is then cut such that the exact number of correct documents still exist within the array, before we finally calculate the precision at that recall.

Within practical 2 we introduce the concept of user feedback through the *Rocchio* algorithm. In order to do this we amend the query method in two ways. Firstly, we introduce the concept of vector weight to each query term, thus whenever a term's score is being appended to a document, its values become a factor of the term's query weight. Secondly, we built our *Rocchio* algorithm. The algorithm creates a new query which additionally includes all terms from our feedback documents. Each term is then iterated over, and a new vector weight calculated, taking into account the *alpha*, *beta* and *gamma* constants. Thus, when a query *q* is entered, a new query and its constituent weights is generated using the *Rocchio* algorithm. This is then used within the updated query *cosine* algorithm. It is also important to note that when calculating precision at recall, we remove the feedback documents from our search space, as these are no longer relevant. If they were to be used in evaluation they would inevitably skew the results.

Results

For practical 1, we achieved the following precision at recalls:

Query 1		Query 2	
Recall	Precision	Recall	Precision

0.09	0.158	0.09	1.0
0.19	0.2	0.19	0.86
0.31	0.26	0.31	0.90
0.41	0.24	0.41	0.93
0.5	0.16	0.5	0.8
0.59	0.14	0.59	0.59
0.69	0.13	0.69	0.5
0.81	0.09	0.81	0.46
0.90	0.08	0.90	0.40
1.0	0.09	1.0	0.32
Average	0.16	Average	0.68

For practical 2, we achieved the following precision at recalls:

Query 1		Query 2	
Recall	Precision	Recall	Precision
0.11	1.0	0.11	1.0
0.19	0.83	0.19	1.0
0.30	0.89	0.30	0.889
0.41	0.65	0.41	0.58
0.52	0.70	0.52	0.61
0.59	0.67	0.59	0.64
0.70	0.61	0.70	0.61
0.81	0.61	0.81	0.65
0.89	0.56	0.89	0.56
1.0	0.36	1.0	0.37
Average	0.69	Average	0.70

Analysis

There were several interesting points noted when evaluating the system, particularly after feedback was introduced:

1. Stemming had little benefit, and in some cases adversely affected out results. This could be down to several reasons such as the fact that our search space was not sufficiently large enough for it to be of benefit. This could alternatively be a quirk of the query, where for whatever reasons, even the stemmed terms fail to appear a sufficient number of times within the expected documents.
2. Down-casing had little or no effect on results, perhaps a result of the fact that the only capitalised query term, "American", should never appear down-cased within professional writing.
3. We found that normalising the document scores with their length had either a slight adverse or no effect. This can largely be attributed to the fact that the documents size differs little from document to document.
4. Once feedback was introduced, our precision increased dramatically for query 1, despite remaining relatively unchanged for query 2. This is likely a result of the fact that the feedback significantly effects query 1 when introduced through *Rocchio*, introducing terms which help further broaden the search. For query 2 however, the search was already general enough that feedback did very little to benefit it.
5. Interestingly the average precision at recall on an empty search query, with feedback, was 0.69, thus indicating that the query term had negligible impact on our results. This might perhaps have been a result of our alpha values.

Conclusion

On its own, we found the *cosine* algorithm to be of little use, with remarkably low precision at recall scores. However, with the introduction of feedback, we found that a general user query performed much better, making it a viable system. One point of criticism for our updated feedback system, might be the extent to which *Rocchio* overrides the initial query, in many ways rendering it effectively redundant.