Result re-ranking

dpj482 - Christian Edsberg Møllgaard

ABSTRACT

In this paper we will peform reranking of search results based on an intra-document methodology. This methodology does not only take term frequency into account, but also tries to use the relative importance of terms in context to individual documents. The methodology used is proposed by Woo et. al[**?**]. Three datasets are used to evaluate the reranking against the original ranking. Here i find no evidence of an improvement and in most cases get worse results than before.

1. INTRODUCTION

With the growing amount of documents on the web, precise and fast searches becomes even more important. In many traditional models, they make use of simple statistics to rank documents relevant to searches. I have tried to re-rank a traditional model to provide better precision.

Other studies[4] have already shown this to be effective, and I want to further explore how this improves searches of on other data sets.

The reason for doing this is, that high precision searches come at a high computational cost. So a simple and fast method is used to get the top results. The top rankings then get re-ranked based on two other models, which we introduce later on.

The model we propose, is based on tf-idf. It determines rank based on the amount of occurrences of a term in a document, and thus provides a query a higher chance of getting ranked the highest. This very simple model has been effective in many other experiments. The tf-idf does however not consider how the document is build. For instance a query x would rank two documents $abxxc$ and $axbxc$ the same. However the second document would have a higher chance of the document being about x, as the mentions of x is spread out through the document, and not just mentioned several times in the same sentence, which could be the explanation in document one.

The two models we need, needs to satisfy the following hypothesises:

- When terms of a query are ranked relatively higher for a document, there is a chance that the query is highly relevant for the document
- If terms of a query are ranked close to each other, they probably describe the contents of the document, which implies higher relevance.

1.1 Other experiments that could influence my result

The great thing about this method is, that it works very well alongside other models. It is possible to combine this method with many other models. As an example you can look at the research found here[2] where they explore how to optimize the tf-idf part. These two papers explore how to tune the weights used when calculating tf-idf. Implementing this, you would increase the effectiveness of the first ranking, which could then be improved upon using our re-ranking method.

According to the articel[2] they gained about 10% rank precision by addressing the problem, that most models use a static weights when calculating their ranks. These static weights are great for specific sized documents and such, but they grow more imprecise the longer the document is from the ideal document size. To handle this they implemented a weight scheme, where they have a weight that is great for short documents and a weight for long documents. They then combine these two weights based on the documents, so they get a more balanced weight regardless of document size.

Another article[1] makes use of weighing terms differently depending on how the affect the query. They show that not every term in the query should be weighted the same. The example they use in the article is:
Consider the following queries with the same term **effect**.

- How does Doppler ultrasound take advantage of the Doppler effect to create a moving image of the inside of the body?
- Effect of temperature on measurement of alkaline phosphatase activity

Effect is the core term when a human reader interprets the query, but the computer will usually weight it equally with all the others.

They have thus implemented a method to check if a term should be weighed higher or lower than the others. This

Information Retrieval 2016 DIKU, Denmark
.

have given them -1% to 26% better results, and according to them does not increase the computational cost alot, which indicates that this could could also be implemented side by side with our term placement re-ranking.

1.2 Another take at re-ranking

Another article[3] have taken another look at re-ranking and figured another way to use it. They have implemented a system where users can personally reconfigure the term weights of searches. This way they can themselves influence their own search results based on the current result. It is mostly and experimental system, but it can provide information about how users really want their searches weighed.

What they do is that they initially just pass the query and provides default answers. The user is then able to manually handle terms, so they get just the results they are looking for, without trying to redo the search.

2. DATA SETS AND QUERIES

This article makes use of three publicly available data sets with documents, queries and qrels (optiman results for all queries).

For the documents data sets it looks like this:

Table 1: document data sets

Name	laTimes	fbis	ft
document count	131896	130471	210158
average length	502	504	399
minimum length	2	12	11
maximum	24125	143175	26145

for the queries I use it looks like this:

Table 2: queries data set

	query count	average length
queries	150	19.54

These are the data sets used for the experiments in this assignment.

3. EXPERIMENTAL SETUP

The experiment is carried out using various tools, where the most important are: Indri, Python 2.7, c++ and TREC_EVAL. To replicate the experiments follow the guide in the appendix.

4. EXPERIMENTS

As a baseline for the experiment, LM with dirichlet smoothening was used. 20 or 50 documents are retrieved for each query, and to tune the dirichlet smoothening, μ is set to $1000, 1500, ..., 3000$ to see when it peforms best for each dataset and query return count. The tuning is done using P_5, and the results are shown in table 3. The tuning of μ turned out to be identical for both query count 20 and 50. The best μ is reported in bold.

The next step is to calculate the two heuristics used in the reranking using the μ values calculated before. The heuristics called R1 and R2, are then used to rerank the querys. The result is then reported using TREC_EVAL with in several different situations. These results are shown in table 4 for query count 20 and in table 5 for query count 50.

μ	μ1000	μ1500	μ2000	μ2500	μ3000
LATIMES	0.2907	0.2920	0.2960	**0.2973**	0.2947
FT	0.3221	**0.3248**	0.3195	0.3221	0.3235
FBIS	0.2581	**0.2608**	0.2554	0.2554	0.2486

Table 3: P@5 values used to tune the experiment

name	type	lm	+R1	+R2	+R1+R2
la	P_5	**0.5137**	0.4104	0.4372	0.4402
la	MMR	**0.2947**	0.2333	0.2427	0.2427
la	NDCG@5	**0.3195**	0.2414	0.2544	0.2547
ft	P_5	**0.5422**	0.4386	0.4198	0.4322
ft	MMR	**0.3235**	0.2483	0.2483	0.2564
ft	NDCG@5	**0.3497**	0.2657	0.2586	0.2672
fbis	P_5	**0.4011**	0.3374	0.3127	0.3307
fbis	MMR	**0.2486**	0.2095	0.2068	0.2108
fbis	NDCG@5	**0.2664**	0.2148	0.2085	0.2134

Table 4: Reranked results for count 20

name	type	lm	+R1	+R2	+R1+R2
la	P_5	**0.5143**	0.3676	0.3872	0.4084
la	MMR	**0.2947**	0.1867	0.1893	0.1933
la	NDCG@5	**0.3195**	0.2014	0.2068	0.2153
ft	P_5	**0.5430**	0.4031	0.3935	0.3900
ft	MMR	**0.3235**	0.2255	0.2295	0.2255
ft	NDCG@5	**0.3497**	0.2396	0.2399	0.2349
fbis	P_5	**0.4026**	0.2810	0.2893	0.2906
fbis	MMR	**0.2486**	0.1730	0.1703	0.1716
fbis	NDCG@5	**0.2664**	0.1742	0.1765	0.1752

Table 5: Reranked results for count 50

5. DISCUSSION

Based on the results presented in table 4 and table 5, it is not possible to claim to say that the the experiment shown by Woo et al.[?] have been reimplemented. Rather the results show a decrease in the score. This could easily be due to a misunderstanding on my part as there is a few things that I am not sure I do correct.

5.1 Possible errors

I am not sure I handle the rankings of TF-IDF correct, if no terms are present in the dataset. This creates a special situation, where I have decided to set it to what i assume is the worst possible score, 0.

6. CONCLUSION

I have tried to reimplement the findings done by Woo et al.[?], but my experiments show a significant decrease in all scores I have experimented with.

7. REFERENCES

[1] D. W. O. Jiaul H. Paik. A fixed-point method for weighting terms in verbose informational queries. ACM, 2014.

[2] J. H. Paik. A novel tf-idf weighting scheme for effective ranking. ACM, 2013.

[3] K. T. Takehiro Yamamoto, Satoshi Nakamura. Rerankeverything: A reranking interface for browsing

search results. pages 1913–1916. ACM, 2011.

[4] H.-W. Woo, J.-T. Lee, S.-W. Lee, Y.-I. Song, and H.-C. Rim. Achieving high accuracy retrieval using intra-document term ranking. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 885–886. ACM, 2010.