Report for Information Retrieval Assignment 2

Mircea Mironenco
University of Amsterdam
MSc. Artificial Intelligence
mircea.mironenco@student.uva.nl

ABSTRACT

Ranking of query results is one of the fundamental challenges of Information Retrieval. In this report we present several ranking algorithms, analyze their performance and draw conclusions regarding the appropriate use-case and robustness of each model, given a fixed dataset of documents and queries.

General Terms

Information Retrieval, statistial language models, term weighting, TF-IDF weighting, language model smoothing, Distributional Semantics

1. INTRODUCTION

Given a query q and a collection D of documents that match the query, a ranking algorithm has to sort the list of documents according to several criterion, such that the results presented to the user are the most appropriate, by some predetermined metric.

We perform several experiments using a variety of ranking models, with the hope of identifying key contexts related to the query and document information, where their application would be ideal.

2. METHODOLOGY

Each model is assessed in accordance to several metrics using the TREC eval utility. Using a dataset of query items, we have query-document pairs that are known to be relevant or not. Given a query and a document, a ranking model will assess the similarity between the query and the document, ranking it higher or lower accordingly.

2.1 Evaluation metrics

After the model outputs the similarity scores for all query-document pairs, we compare them to the **ground truth** and consequently score the algorithm using the following metrics:

1. Normalized discounted cumulative gain at rank 10 (NDCG@10). Measures the performance of the ranking system based on the graded relevance of the documents.
2. Mean Average Precision. The mean of the average precision score for each query. Particularly we are interested in the MAP at rank 1000.
3. Precision at rank 5. The fraction of documents retrieved that are *relevant* to the query, and implicitly the user's specific information.
4. Recall at rank 1000. The fraction of documents that are relevant to the query, and successfully retrieved.

2.2 Statistical significance

Furthermore, after the results are obtained, we study the statistical significance between the similarity scores obtained using different models.

Given a set of results, we perform 4 comparisons between the algorithms, for each evaluation metric used. The test measures whether the average score differs significantly across samples. Our *null hypothesis* is that the averages are the same.

We solve the multiple comparisons problem using ÅăidÃąk correction [5], which is a method to control method to control the familywise error rate. Respectively, we set a value of α, and reject each null hypothesis that has a p-value lower than $alpha_{SID} = 1 - (1 - \alpha)^{\frac{1}{m}}$ where m is the number of null hypotheses we are testing.

3. EXPERIMENTAL SETUP

3.1 Research Questions

The experiments are divided into 2 categories, corresponding to the families of ranking methods that are being tested, *lexical models* and *latent semantic models*.

The set of experiments are performed as follows. Given the set of documents and queries, we construct basic ranking models. Some research questions we aim to answer are:

- Given the same set of documents and queries, which Lexical or Semantic retrieval methods perform better, given the previously defined metrics.
- How robust each model is given common issues encountered with various parameters, such as document length, corpus size, frequency of words, etc.
- Are the differences between the performance of the models statistically significant.
- What are the best ways to tune the models which incorporate a set of hyperparameters.
- Can incorporating a more robust probabilistic framework, or various language models enhance the relevance of a ranking generated by a more basic model, such as TF-IDF.

3.2 Model implementation

In this section we discuss the models which have been implemented and what changes have been made to the standard implementations, to speed up the experiments. Unless otherwise explicitly stated, models are trained on the entire document collection, and the top 1000 documents for each query are saved.

TF-IDF.

The implementation has been modified to account for the size of the document when computing the term frequency component. Respectively we use *augmented term frequency*, which normalizes the term frequency weights of a document, by the maximum frequency occurring in that document.

BM25.

Okapi BM25[6] has a standard implementation using $k_1 = 1.2$ and $b = 0.25$.

Smoothing models.

Models which aim to smooth the distribution of the words (Jelinek-Marcer, Dirichlet Prior, Absolute Discounting) have a standard implementation. The parameter optimization has been done using grid-search on the validation set. The best value was chosen by considering the best performance using NDCG@10.

Positional Language Models.

presented several difficulties because of the expensive operations performed, and several optimizations have been made:

- PLMs[3] are used only for re-ranking the top 400 documents retrieved using TF-IDF.
- For each kernel we compute both the Cumulative Distribution Function and a max_doc_length x max_doc_length matrix, to perform block-wise operations on. The CDF and matrix operations give us constant time access to values which are then cached to be reused.
- Documents with under 350 words are removed from the ranking, both for time complexity and to test the robustness of the count propagation assumption.
- The 'tails' of the documents are cut off, and PLM are calculated only from the first and last occurring query element. This does not affect ranking.

Distributional semantic models.

All distributional semantic models have been implemented, the following optimizations being made:

- Latent Semantic Indexing[4] uses 64 topics. While performance most likely was affected this is done to speed up the experiments.
- Similarly, word2vec and doc2vec[2] use 32 units in the hidden layer, and Latent Dirichlet Alocation[1] uses 32 topis.
- All 4 distributional semantic models are trained on the entire corpus, but only used to rerank the top 1000 documents retrieved using the TF-IDF algorithm
- Document and query representations are constructed using averaging once each word has been transformed into the latent space.
- Cosine similarity is used to measure the level of similarity between the query and the document.

3.3 Other implementation notes

All experiments are done in Python in a Jupyter notebook, which can be found inside the zip file together with this report. Alongside them is a file called *requirements.txt* which will indicate which packages were installed on the system at the time the assignment was done (Note that some packages may not be part of PyPi). Furthermore, there are also 2 shell scripts used when evaluating the runs produced using *Trec eval*. The third task of the assignment has not been completed fully, and has been lef out of the notebook.

Experiments can be run from the first cell to the last, without interruption however to run the experiments, the shell scripts are necessary as they are used by a class which interacts with the Trec eval utility. The reader also has to keep in mind that there are multiple sections that are computationally intensive, both in terms of time and memory. The results of the experiments, are continually saved to disk, and loaded when necessary. Furthermore, there are several instances where all the memory used is cleared, in these cases, some of the notebook cells need to be rerun in case the algorithms they define are to be reused.

4. RESULTS AND ANALYSIS

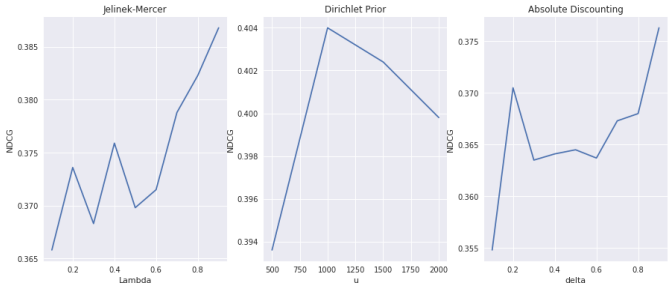
Following are the results of the semantic and lexical models on the test set.

We discuss the results in terms of the initial research question that were posed in a previous section.

Table 1: Test set results for all constructed models.

	NDG@10	MAP@1000	P@5	R@1000
TF-IDF	0.3428	0.1829	0.3567	0.6089
BM25	**0.4109**	**0.2183**	**0.4217**	**0.6493**
Best-Jelinek	**0.3849**	**0.1977**	0.375	**0.6288**
Best-Dirichlet	**0.4139**	**0.2119**	**0.4283**	**0.6313**
Best-Abs. D.	**0.396**	**0.2037**	0.4017	0.6258
Best P.L.M.	0.3708	0.137	0.3733	0.3708
Word2Vec	0.3113	0.1601	0.3083	0.6089
Doc2Vec	**0.1042**	**0.0741**	**0.1100**	0.6089
LSI	**0.0159**	**0.0485**	**0.0133**	0.6089
LDA	**0.0159**	**0.0413**	**0.0167**	0.6089

Figure 1: NDCG values for smoothing models with various parameter values



We can observe that the *best performing models* are BM25 and the parameter-optimized Dirichlet Prior smoothing model. The distributional semantic models have the worst performance. This can be explained by the low dimensionality chosen for each model to optimize the speed of the experiments.

The smoothing models account for the infrequency of the words and model both a document language model and a collection language model. These models are optimized using grid-search, their hyper-parameters being chosen for the best performance on the validation set.

The BM25 model, presents a probabilistic interpretation of the term frequency and inverse document frequency, and has shown to be robust and adaptable taking into account both the size of the document, the collection and the frequency of words.

The Positional Language Model, which is used to simply re-rank the TF-IDF retrieved documents improves the DCG measure. The MAP@1000 and Recall@1000 are not so relevant here, as we have less than 400 documents retrieved by this model per query.

Statistical significance can also be deduced from Table 1. All models are compared against a baseline model, the TF-IDF algorithm, using all 4 measures. The bolded values are values for which the null hypothesis (identical average scores) has been rejected. Note that these results also account for the correction done to mitigte the multiple comparisons problem.

Hyper-parameter tuning using grid-search is slow and problematic. In this situation one can use a binary search approach where 2 extreme values are chosen and half of the search space is reduced in favor of the better performing one. An alternative way of computing the parameters, at least in the case of the smoothed language models is by using a two-stage smoothing model, which uses the EM algorithm to optimize it's parameters.

Regarding the last research question, indeed probabilistic frameworks if correctly tuned can capture more meaningful representations of the data, and make decisions that have a stronger mathematical support, in comparison to very simplistic geometric models such as TF-IDF.

5. CONCLUSIONS

We have experimented with several categories of retrieval models, with the aim of testing their performance on a given set of documents.

Smoothed models have numerous advantages both in terms of the probabilistic properties they express and their ability to deal with datasets that are not ideal for document ranking. However, their performance is very dependent on the hyperparameters chosen.

Distributional semantic models, and models that use neural networks require vast amounts of data to be trained properly. Moreover, in our case we reduced the dimension of the representations to perform the experiments faster. This most likely caused the models to underfit, as the complexity of the model was reduced. Under the frequentist perspective of bias-variance trade-off the model has a high bias and was unable to capture the semantic or syntactic properties in the corpus.

Language models are promising because they have been extensively studied in the context of language estimation for natural language processing tasks. Neural language models have proved to be extremely powerful, being able to easily capture both semantic and syntactic information using only vector representations and backpropagation.[**?**]

6. REFERENCES

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *JLMR*, 2003.

[2] Q. Le and T. Mikolov. Distributed representations of sentences and documents. 2014.

[3] Y. Lv and C. Zhai. Positional language models for information retrieval. 2009.

[4] G. W. F. T. K. L. Scott Deerwester, Susan T. Dumais and R. Harshman. Indexing by latent semantic analysis.

[5] Z. K. Sidak. Rectangular confidence regions for the means of multivariate normal distributions. *Journal of the American Statistical Association.*, 1967.

[6] S. W. Sparck Jones, Karen and S. E. Robertson. A probabilistic model of information retrieval: Development and comparative experiments. 2000.