Natural Language Processing Project

Kaushik Arcot and Mohana Sai Krishna

Indian Institute of Technology Madras

Abstract. In this project, we tried out different NLP methods to implement a search engine on the Cranfield Dataset. Our main focus was on introspective methods, to improve on the baseline of the vector-space based model, which used a TF-IDF encoding of the document-word matrix and calculated similarity using cosine of the query and document vectors.

Keywords: LSA · pLSA · LDA · BM25

1 Introduction

The Cranfield Dataset is a small curated corpus of 1400 document. Then, there are 226 queries on the dataset, where each query is a string of words. This dataset is extensively used in Information Retrieval Experiments.

2 Problem Definition

For each query, there are a set of documents that are relevant and there are scores that are given to each document, 1 for the most relevant document, to 4 for the least relevant document. The task is to retrieve documents for each query, and we're looking at two measures, MAP and nDCG, which give weightage to the position of the document and the relevance score for that document.

3 Baseline

Our first assignment in the NLP course was to build a simple search engine using a vector-space based model. Each document and query was passed through preprocessing steps, involving sentence segmentation,tokenization, lemmatization, stop word removal. Then, each document is represented as a vector in the space of words, using a TF-IDF transformer. Then, the similarity of a query with a document is measured as the cosine of the angle between their respective vectors, and they're sorted based on this similarity measure.

Here are the different measures for this approach, at different levels of retrieval, i.e retrieving the top [1,2,3....,10] documents for the query.

For example, for k = 10, these were the metrics:
Precision, Recall and F-score @ 10 : 0.29555555555555585, 0.4224968243187276, 0.3216174756733598
MAP, nDCG @ 10 : 0.6695642017300746, 0.48720270205172517

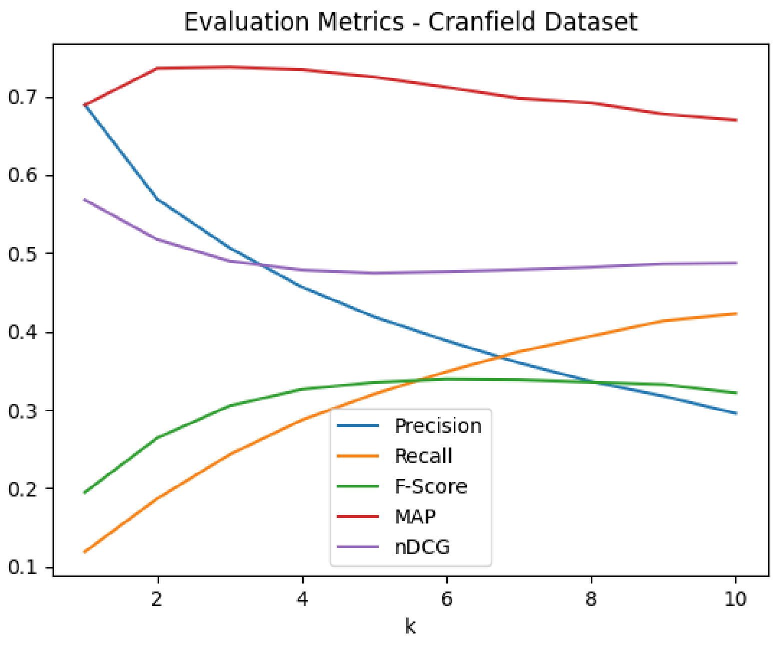


Fig. 1. Evaluation Metrics for TF-IDF Vector space based search engine model.

4 Motivation

We had observed several points of improvement in the approach, in the different steps of pre-processing, document encoding and query document similarity:

- On observing the Cranfield Dataset, we found that there were many hyphenated words, that were not split as separate words by the Penn-Treebank Tokenizer, even though the two words on either side of the suffix made sense independently and were relevant to the document.
- Several words in the documents were just simply numbers or numbers that were attached to a concept, and were irrelevant to simple text-based search system. Eg:'431','1.23' are irrelevant to a document on their own as a token, with respect to simple queries.
- A vector space-based model does not consider similarity between words that convey similar meaning. It only considers two documents similar if they have the same words, without looking into the semantics of the document, or the underlying meaning or topic of the document.

To address these problems, we tried the following steps:

- We built our own tokenizer, using a simple delimter based string splitter. Different special characters are considered as delimiters and the string is broken up into a list of tokens.
- We removed number tokens from the document as well as from the query, and also removed very small tokens.
- We concentrated on introspective NLP methods to improve the representation of a document, and thus improve the measure of similarity to bring it closer to the human standard of similarity judgement.
- To make the title more important, we added the title of the document to the body of the document. Because the title is generally the first sentence of the document (for this dataset), our operation basically doubled the importance of the title, with respect to the body of the document

5 Methodologies

5.1 Latent Semantic Analysis

We can understand this technique by the name. In this method, we look to extract "latent" topics in the document space. LSA assumes the distributional hypothesis, i.e LSA assumes that words that are close in meaning will occur in similar pieces of text. Thus, it uses co-occurrences to similarity between words, and then uses similar words to extract "latent" topics from the document space.

First, we use a document-term matrix as an input to LSA. This can be either a bag of words matrix or a TF-IDF matrix. Then, we apply SVD (Singular Value Decomposition) on this matrix, and reduce the dimensionality of the document space, to a decided number of topics. We can also extract the word distribution across the topics, and using that, we get the representation of the query in the topic space. Say, X is the document-term matrix, we get U,Σ and V. U being the document-topic space matrix, Σ the singular value diagonal matrix, and V the word-topic space matrix.

The number of topics present in the document space is a hyper-parameter.

$$X = U\Sigma V^T$$

5.2 Probabilistic Latent Semantic Analysis

pLSA is a generative model on the set of documents that it has estimated on. It arrives at the probability of the document-word co-occurence as a mixture of multinomial distributions on the topic in the discrete document space, and then the word in the subsequent topic space. The number of topics is a hyper-paramter for the model, as in LSA and in LDA.

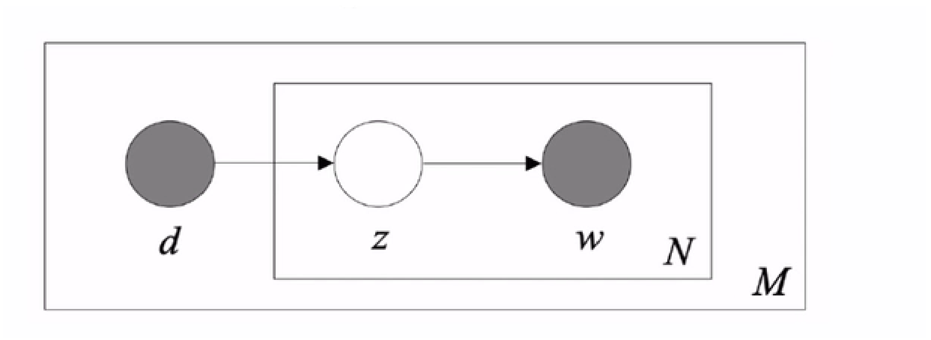


Fig. 2. A visualization of the technique used in pLSA

$$p(d,w_n) = p(d)\sum_{z} p(w_n|z)p(z|d).$$

Fig. 3. The formula for a document-word co-occurence probability in pLSA

5.3 Latent Dirichlet Allocation

From pLSA, we tried a new introspective approach, known as LDA. It is a generative approach, i.e it is a model to generate new documents, that involves Dirichlet multinomial distributions for the topic-word distributions and document-topic distributions. (i.e Distribution of Distributions) It is extension of pLSA where the documents are spread out in a continuous space of topics, rather than in a discrete space, as seen in pLSA.

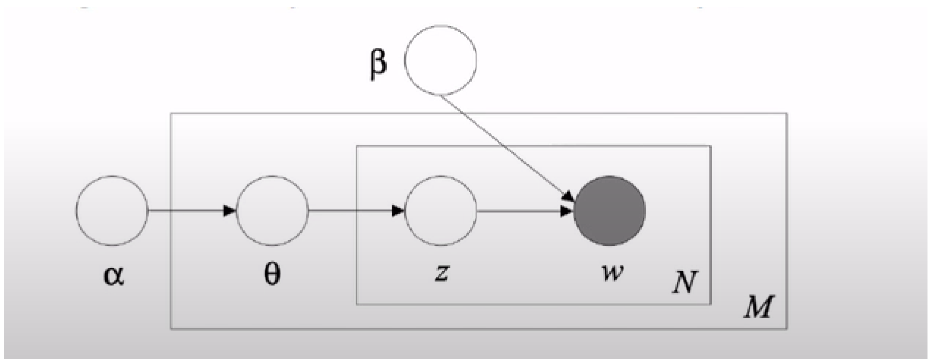


Fig. 4. An understanding of the LDA method. α and β are Dirichlet distributions, whose output are the document-topic distribution and the topic-word distribution respectively

$$p(\mathbf{w} \mid \alpha, \beta) = \int p(\theta \mid \alpha) \left(\prod_{n=1}^{N} \sum_{z_n} p(z_n \mid \theta) p(w_n \mid z_n, \beta) \right) d\theta.$$

Fig. 5. Probability of a document, using the LDA approach

6 Other Methodologies Tried

We also came across a statistical measure, called Okapi BM-25, which is a scoring method tailored to a search engine implementation. This score gives a ranking of the corpus documents given a particular search query. The score of a document D given a query Q is:

$$score(D,Q) = \sum_{i=1}^{n} IDF(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{avgdl}\right)}$$

where $q = (q_1, q_2, \ldots, q_n)$ are the n words in the query. $f(q_i, D)$ is the frequency of q_i in document D, k_1, b are hyper-parameters, and $|D|$ is the length of document D (number of words), and $avgdl$ is the average length of a document in the corpus. $IDF(q_i)$ is similar to the inverse document frequency score in TF-IDF

$$IDF(q_i) = \ln\left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1\right)$$

We studied the mathematics of this ranking method on the **Wikipedia Page** and the research and results of this method at this **source**

7 Experiments and Results

Here are the evaluation plots for the different methods tried.

We tried to set a different baseline by employing the TF-IDF vector space-based method to the pre-processing steps we came up with. Given further below is a table of MAP and nDCG scores for the different methods tried

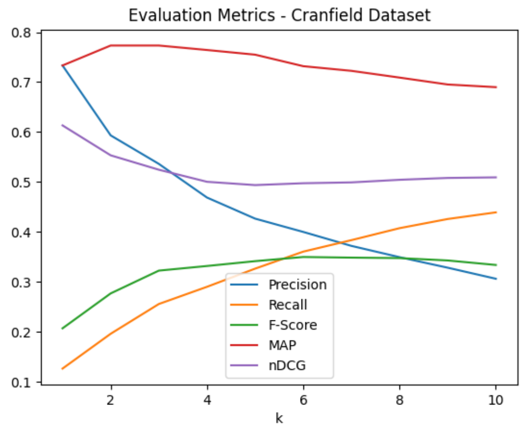


Fig. 6. Metrics for updated TF-IDF method

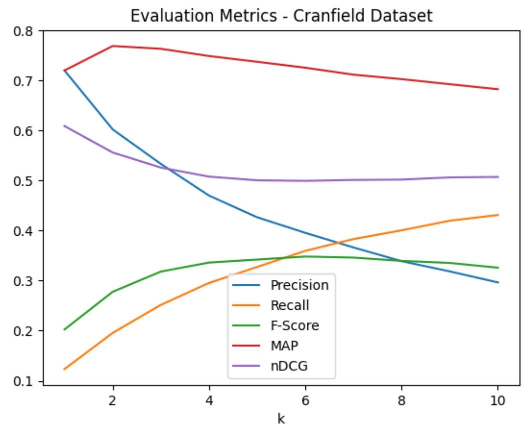


Fig. 7. Metrics for the BM25 measure method

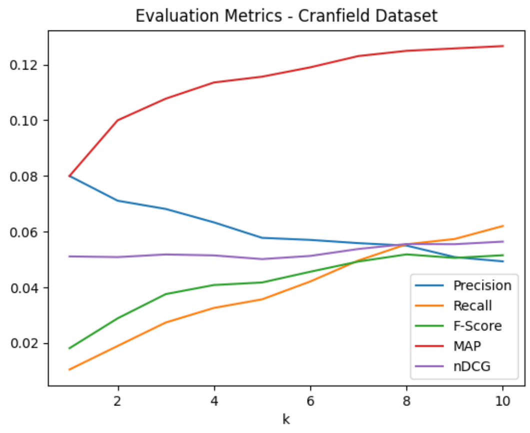


Fig. 8. Metrics for the LSA method

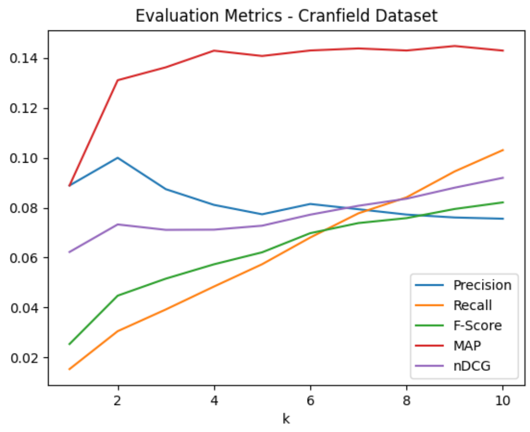


Fig. 9. Metrics for the pLSA method

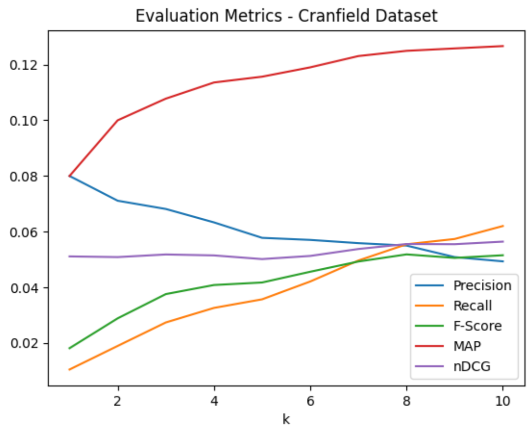


Fig. 10. Metrics for the LDA method

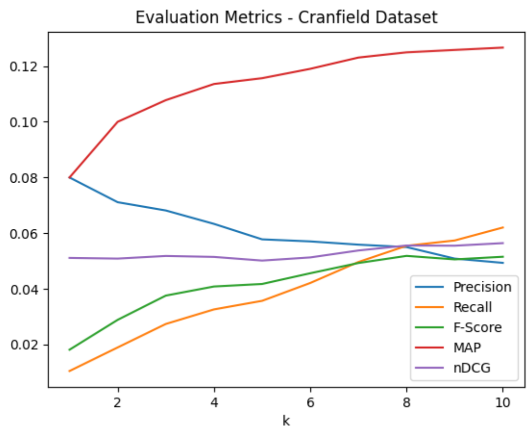


Fig. 11. Metrics for the LDA method

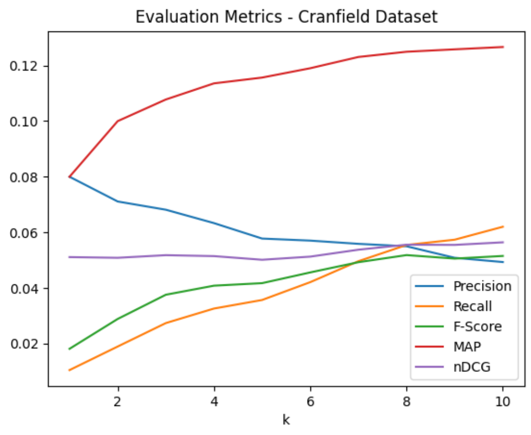
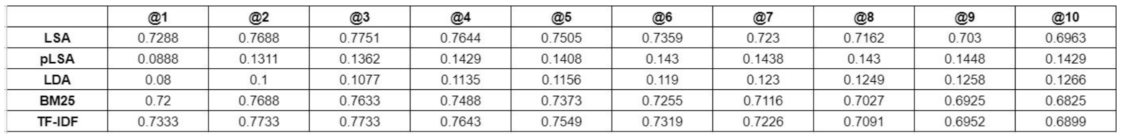
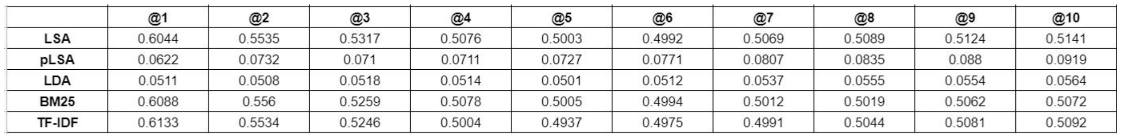


Fig. 12. Metrics for the LDA method



	@1	@2	@3	@4	@5	@6	@7	@8	@9	@10
LSA	0.7288	0.7688	0.7751	0.7644	0.7505	0.7359	0.723	0.7162	0.703	0.6963
pLSA	0.0888	0.1311	0.1362	0.1429	0.1408	0.143	0.1438	0.143	0.1448	0.1429
LDA	0.08	0.1	0.1077	0.1135	0.1156	0.119	0.123	0.1249	0.1258	0.1266
BM25	0.72	0.7688	0.7633	0.7488	0.7373	0.7255	0.7116	0.7027	0.6925	0.6825
TF-IDF	0.7333	0.7733	0.7733	0.7643	0.7549	0.7319	0.7226	0.7091	0.6952	0.6899

Fig. 13. MAP scores for different methods tried



	@1	@2	@3	@4	@5	@6	@7	@8	@9	@10
LSA	0.6044	0.5535	0.5317	0.5076	0.5003	0.4992	0.5069	0.5089	0.5124	0.5141
pLSA	0.0622	0.0732	0.071	0.0711	0.0727	0.0771	0.0807	0.0835	0.088	0.0919
LDA	0.0511	0.0508	0.0518	0.0514	0.0501	0.0512	0.0537	0.0555	0.0554	0.0564
BM25	0.6088	0.556	0.5259	0.5078	0.5005	0.4994	0.5012	0.5019	0.5062	0.5072
TF-IDF	0.6133	0.5534	0.5246	0.5004	0.4937	0.4975	0.4991	0.5044	0.5081	0.5092

Fig. 14. nDCG scores for different methods tried

8 Conclusion

As we can see from the nDCG scores, LSA (0.5141) performs better than the original baseline of TF-IDF (0.4872) and also the improved baseline for TF-IDF (0.5092). The same can be seen for the MAP scores (0.6963 vs 0.6696, 0.6899) for k = 10 (10 retrieved documents). This same trend is seen for lower k values too.

Now, for pLSA and LDA, the scores are much lower. At k = 10, nDCG scores LDA : 0.0564 pLSA : 0.0919

This shows that deeply introspective methods with many parameters to estimate do not work well with the Cranfield Dataset, with it's low document count of 1400. With a corpus of the order of 100,000 documents, and varied topics to extract from the corpus, pLSA and LDA could perform better on these metrics. A simple introspective method like LSA, that is only concerned with Singular Value Decomposition and Dimension Reduction works well with the corpus. pLSA and LDA are also computationally very expensive methods to implement, with pLSA taking approximately 3 hours to complete the distribution estimation, and similar time periods for LDA. In this aspect, LSA performs on a similar timescale to the TF-IDF measure method (10 minutes). Overall, LSA is a better implementation of a search engine than TF-IDF based search engine on the Cranfield dataset, based on MAP and nDCG scores.

Now, as a statistical method, BM25, as an alternate to TF-IDF, performs about equally on the measures At k = 10, nDCG scores
TF-IDF : 0.5092 BM25:0.5072

MAP scores: TF-IDF : 0.6899 BM25 : 0.6825

Overall, we discovered that introspective methods that look to extract latent information from the corpus of documents do not provide a massive improvement on the vector-space-based TF-IDF model. There are other approaches that can be used, an example could be a knowledge-based method, like ESA. This could improve performance by using the vast amount of articles present on Wikipedia to essentially encode a concept as a vector of words related to that concept. We in this project have explored the depth of introspective methods, working on nothing else but the corpus and the queries given. LDA was a new approach we discovered, which was an extension of the concepts learned in pLSA in the course. And we also came across the Okapi-BM25 measure, meant for a query-corpus based ranking search engine.