CS6370 - NATURAL LANGUAGE PROCESSING

Purvam Jain - EE20B101 and Sridurgaa D V - CH19B091†

Contributing authors: ee20b101@smail.iitm.ac.in;
ch19b091@smail.iitm.ac.in;
†These authors contributed equally to this work.

1 Introduction

The aim of this project is to improve the TF-IDF informational retrieval system used in Assignment-2 on the Cranfield dataset. We have introduced spell check into engine and tried Latent Semantic Analysis and BM-25 models.

2 Spell Check and correction

We used a basic tf-idf representation of words, some of which are misspelled having their own dimension in the vector space. This has a huge impact on the IR system's performance. We tried using 2 ways to identify and correct the spellings.

2.1 PySpellChecker:

PySpellChecker is designed to be easy to use to get basic spell checking. It uses a Levenshtein Distance algorithm to find permutations within a particular edit distance from the original word. It then compares the generated permutations (Candidates) to a set of known words with associated frequency scores. PySpellChecker uses the word with highest frequency among the generated candidates as the replacement for the original word.

2.1.1 TextBlob:

TextBob library can be used for POS tagging, sentiment analysis, lemmatization, finding synsets, and also spelling correction. Correct() method of TextBlob library is also context insensitive. Hence, it sometimes over-kills to produce new incorrect words.

Both PySpellChecker and TextBlob gave similar results. The results were comparable to the VSM based IR system. Comparing plots of TexBlob and VSM based IR shows that all metrics have similar values for all k values and there is very small reduction in the MAP score for higher values of k. This could be attributed to the fact that most of the words in our corpus were correct and there was little impact just by changing some words on the performance of our IR system.

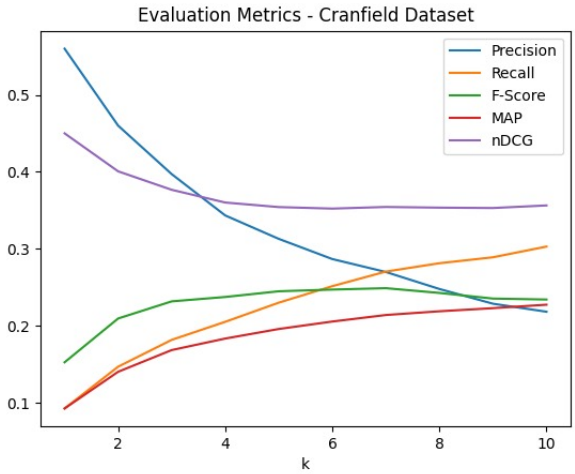

Fig. 1: Evaluation Metrics for IR system with TextBlob based spell correction

3 Latent Semantic Analysis

We implemented lsa along with spell check and checked if the performance improved. When compared to the basic model used in the assignment, the performance improved very marginally and was not very significant. The results obtained are shown below.

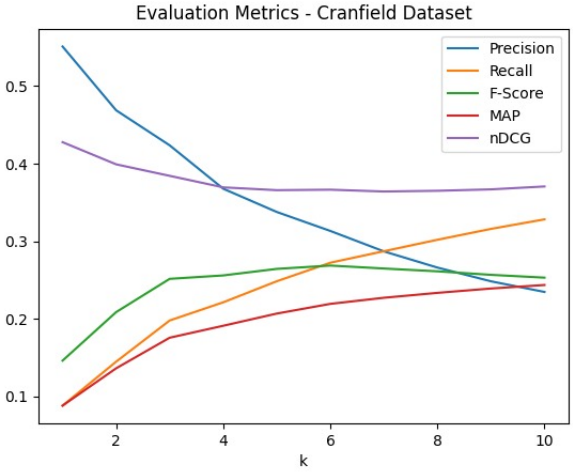

Fig. 2: Evaluation Metrics for IR system with LSA

The IR system with LSA, Recall, MAP, F-score have higher and better values for a range of k values when compared to VSM model. Since, word representations from LSA capture context and information better than orthogonal vectors, IR system retrieves higher number of related documents even without the same exact words as expected. Hence, using LSA improves performance of the system. When pyspellchecker was used for spell correction along with LSA, the IR system performed very poorly compared to others. This is because, pyspellchecker replaces words with highest frequency word from the generated candidates. This might replace correct infrequent words with irrelevant frequent words and hence changes the meaning of queries and documents entirely.

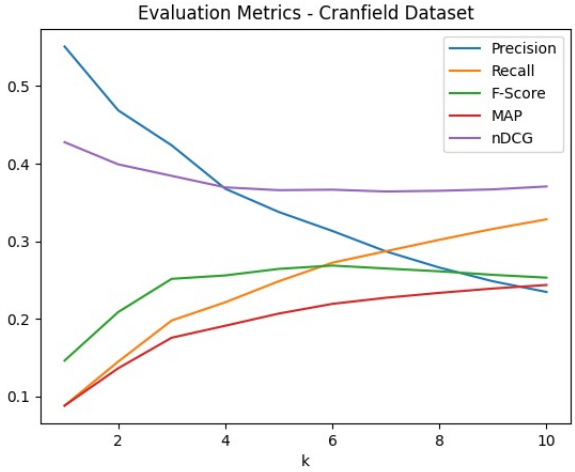

Fig. 3: Evaluation Metrics for IR system with PySpellChecker and LSA

This plot is for TextBob based spell check during pre-processing and then LSA is used during indexing. Using spellcheck along with LSA has a better performance than VSM for Recall, F-Score, MAP. But when compared to the IR system using only LSA, there is very less improvement for MAP. The reson is maybe spelling correction corrects the incorrect words both from queries and documents, thereby documents which will have same words will be retrieved and hence gives higher precision for the IR system.

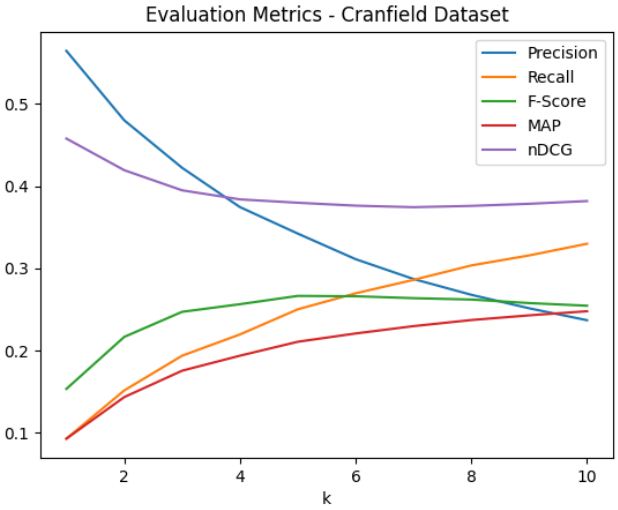

Fig. 4: Evaluation Metrics for IR system with TextBlob and LSA

4 BM25

BM25 is a ranking function that ranks a set of documents based on the query terms appearing in each document, regardless of the inter-relationship between the query terms within a document (e.g., their relative proximity). It is a family of scoring functions with slightly different components and parameters. One of the most prominent instants of the function is as follows. Given a query Q, containing keywords q1, ..., qn The BM25 score of a document D is:

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

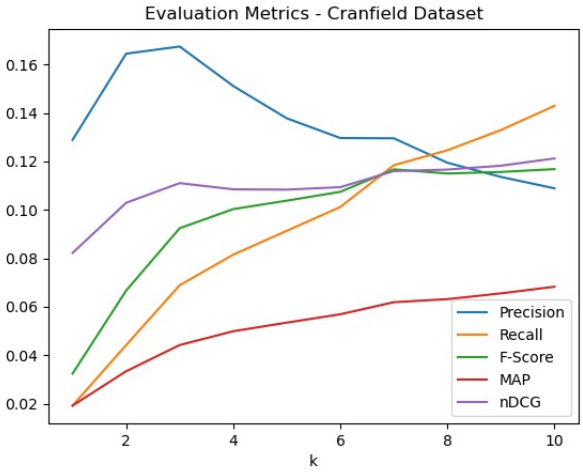

Fig. 5: Evaluation Metrics for IR system with BM25

Above plot is generated for IR system with BM25. With BM25, the system performs poorer than VSM. This might be due to hyper parameter selection.