Experimental Project: search experiments

100036248 or C.A.Irvine

1. INTRODUCTION

Modern Information Retrieval (IR) systems, by definition, are expected to return documents that are relevant to the supplied Search Query. What are the necessary steps to achieving this goal? What tools will be employed? How do we test and improve an IR system?

This paper aims to answer these questions, providing the steps needed to build an accurate IR system. By the end, we will have the knowledge needed to build and continuously improve a retrieval system that can be applied to domains of any size. For the purposes of this paper, we will be building the system using the portal.uea.ac.uk domain.

2. INITIAL SYSTEM

The base IR system can be divided into two sub-systems: indexing and retrieval.

The base retrieval system is not accurate, when a relevant document is mentioned, it means the system believes it to be relevant. In reality, the returned documents are rarely relevant for this base system.

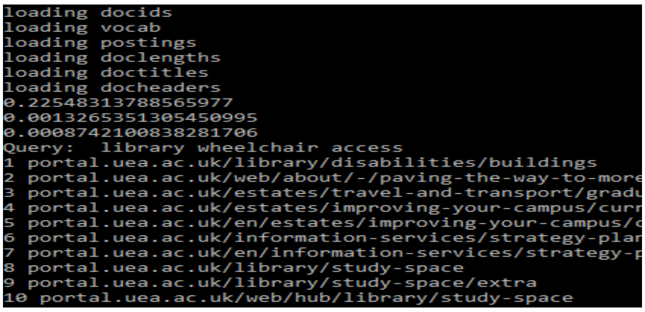
For ease, this system and subsequent systems will print to the command line, allowing the focus of this experiment to remain solely on returning relevant documents, see Figure 1.

2.1 Indexing

The indexing subsystem takes a web domain and uses a **NIST PCcrawler web spider program** ([Pozo, 2010]). Each document is cleaned of all HTML, numbers, special characters and contractions. The 'clean' document text is separated into tokens and from there a vocabulary of unique terms is generated. For every term in the vocabulary; the occurrence of that term is counted for each document, this creates the postings table.

2.2 Retrieval

The retrieval subsystem reads the Search Query supplied by the user, 'cleans' it using the same method as indexing (Section 2.1), and then uses the postings table to generate the **Term Frequency * Inverted Document Frequency (TF*IDF)** score for each word in the query and each document relevant to that word. The scores are combined and all relevant documents are ordered. The top 10 documents are displayed to the user.



```
loading docids
loading vocab
loading postings
loading doclengths
loading doctitles
loading docheaders
0.22548313788565977
0.0013265351305450995
0.0008742100838281706
Query:  library wheelchair access
1 portal.uea.ac.uk/library/disabilities/buildings
2 portal.uea.ac.uk/web/about/-/paving-the-way-to-more
3 portal.uea.ac.uk/estates/travel-and-transport/gradu
4 portal.uea.ac.uk/estates/improving-your-campus/curr
5 portal.uea.ac.uk/en/estates/improving-your-campus/c
6 portal.uea.ac.uk/information-services/strategy-plar
7 portal.uea.ac.uk/en/information-services/strategy-p
8 portal.uea.ac.uk/library/study-space
9 portal.uea.ac.uk/library/study-space/extra
10 portal.uea.ac.uk/web/hub/library/study-space
```

Figure 1: Output produced from the retrieval subsystem, stemming (see Section ?? and weighting (see Section 4.2)is present in this system

3. TF*IDF

As discussed in Section 2.2, the retrieval system uses TF*IDF to rank the relevance of each document. TF*IDF returns a score between 0 and 1, 1 being entirely relevant. The formula for TF*IDF is below (([Singhal, 2001]), ([Manning et al., 2008]):

$$TF * IDF \;=\; (log_{10}\frac{N}{n_{\mathrm{i}}}) \times (\frac{freq_{\mathrm{i,j}}}{P_{\mathrm{j}}})$$

$$freq_{\mathrm{i,j}} = Frequency\ of\ term\ i\ in\ document\ j$$

$$P_{\mathrm{j}} = Number\ of\ terms\ in\ document\ j$$

$$N = Number\ of\ documents\ in\ collection$$

$$n_{\mathrm{i}} = Number\ of\ documents\ containing\ term\ i$$

4. ENHANCED SYSTEM

Taking the system discussed previously (see Section 2) as a base, we can drastically enhance the quality of the retrieved documents with a few simple additions; namely stemming and weighting.

4.1 Stemming

Stemming is the process of removing prefixes and suffixes from words to leave the roots. This allows words to be gener-

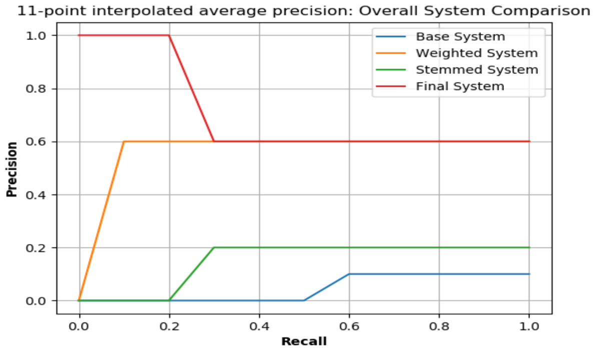


Figure 2: Line graph showing the 11-point interpolated average precision for each stage of the system.

alised and prevents natural speech patterns interfering with both the indexing and retrieval subsystems (as the search queries are 'cleaned' in the same manner as the indexing).

The **Natural Language ToolKit (NLTK)** ([NLTK, 2017]) provides several stemming modules. Each module stem words to different levels. Stemming carries a risk that the word is stemmed incorrectly and will become irrelevant to all searches ([Hull et al., 1996]). Three modules were tested; Snowball, PorterStemmer and WordNet (listed in order to strength).

Three vocabularies were generated, one for each module, for the same 30 documents. Each word that was not a recognisable English word was counted for each module. The lightweight, WordNet Lemmatizer stemmer was chosen, as it had the least amount of incorrect words.

4.2 Weighting

Weighting prioritises documents that mention search terms in Titles, Headers and in the opening paragraph of the document. Further alteration to the indexing subsystem is needed, but also requires the retrieval subsystem is adapted, to implement Weighting.

The indexing subsystem must copy the tokens that are in the **<title>**, **<h>** and **<meta content="">** tags are stored in their own files. In addition to this, the **first thirty tokens** for each document is stored separately as well.

In the retrieval subsystem, these separated files are read in and searched for the Search Query Terms. If they are found in the files an additional multiplier is applied to that term's IDF value (see Section 3).

5. EVALUATION

At each stage of the system's development, the precision was tested by running the same query - "multifaith centre opening times" - and comparing the precision to the base. We can see from Figure 2 that the Final System (which incorporates both Weighting and Stemming) is significantly superior to the base system, however precision does drop and plateau at the Weighted System level.

5.1 Set Query Evaluation

The supervisor of this project supplied a list of 10 queries for the system to be evaluated with. The top 100 returned documents were analysed by a python script was written to generate statistical and graphical analysis on these queries. One of the critical statistics returned by the script was the **Average Precision Value (APV)**. Queries 1, 5, 8 and 10 are presented in Table 1 because collectively they are the best, worst and most inaccurate results produced by the final system. They average is also presented for comparison.

Table 1: Table showing select Average Precision Values (to 3 sf) from a set of 10 Queries, in addition to the Average of the 10 Queries

	Q1	**Q5**	**Q8**	**Q10**	**Average**
Average Precision Value (3 sf)	0.321	0.691	0.929	0.148	0.638

5.2 Q5 Investigation

As we take a closer look into the 68 documents returned by Query 5: **Neil Ward** we start to understand why Q5 had an above average APV. 52 of the 68 documents mention a different Neil within the portal.uea.ac.uk domain, meaning that the majority of returned documents are irrelevant to the User. The first 6 results returned directly concern **Neil Ward**, so the User would likely be satisfied by the returned documents.

To improve on this inaccurate result, **Named Entity Recognition (NER)** tools supplied by NLTK (see Section 4.1, [NLTK, 2017]) can be integrated to the system. This will likely improve other query results.

5.3 Future Improvements

In addition to the NER improvement suggested in Section 5.2, a Graphical User Interface (GUI) could be developed so that the Snippets (mentioned in Section 4.2) can be displayed. The GUI can also provide a platform for User Feedback so that Relevance Feedback can continuously improve the IR System.

6. CONCLUSION

In this paper we have learned how to build an Information Retrieval System (see Sections 2, 3 and 4), using industry standard tools (see Section 4.1). Then we learned how to evaluate the Information Retrieval System (see Section 5), with the aid of a Python Script, looking at Graphs and Statistics to identify areas for improvement within the Information Retrieval System (see Section 5.2). Finally, additional future improvements for the IR System were suggested with the aim of improving both Accuracy and Usability.

7. REFERENCES

[Hull et al., 1996] Hull, D. A. et al. (1996). Stemming algorithms: A case study for detailed evaluation. *JASIS*, 47(1):70–84.

[Manning et al., 2008] Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to information retrieval.* Cambridge University Press.

[NLTK, 2017] NLTK, O. (2017). Natural language toolkit.

[Pozo, 2010] Pozo, R. (2010). Nist web crawler.

[Singhal, 2001] Singhal, A. (2001). Modern information retrieval: A brief overview. *IEEE Data Eng. Bull.*, 24(4):35–43.