

CS6370: Natural Language Processing Project

Release Date: 24th March 2024

Deadline: 8th May 2025

Name:

Roll No.:

General Instructions:

1. The template for the code (in Python) is provided in a separate zip file. You are expected to fill in the template wherever instructed. Note that any Python library, such as nltk, stanfordcorenlp, spacy, etc, can be used.
2. A folder named 'Roll_number.zip' that contains a zip of the code folder and your responses to the questions (a PDF of this document with the solutions written in the text boxes) must be uploaded on Moodle by the deadline.
3. Any submissions made after the deadline will not be graded.
4. Answer the theoretical questions concisely. All the codes should contain proper comments.
5. For questions involving coding components, paste a screenshot of the code.
6. The institute's academic code of conduct will be strictly enforced.

The first assignment in the NLP course involved building a basic text processing module that implements sentence segmentation, tokenization, stemming/lemmatization, stopwords removal, and some aspects of spell check. This module involves implementing an Information Retrieval system using the Vector Space Model. The same dataset as in Part 1 (Cranfield dataset) will be used for this purpose. The project is split into two components - the first is a *warm-up* component

comprising of Parts 1 through 4 that would act as a precursor for the second and main component, where you improve over the basic IR system.

Consider the following three documents:

d_1 : Herbivores are typically plant eaters and not meat eaters

d_2 : Carnivores are typically meat eaters and not plant eaters

d_3 : Deers eat grass and leaves

1. Assuming {are, and, not} as stop words, arrive at an inverted index representation for the above documents.

2. Construct the TF-IDF term-document matrix for the corpus $\{d_1, d_2, d_3\}$.

3. Suppose the query is "plant eaters," which documents would be retrieved based on the inverted index constructed before?

4. Find the cosine similarity between the query and each of the retrieved documents. Is the result desirable? Why?

Cosine Similarity calculations:

Ranking documents:

Is the ordering desirable? If no, why not?:

1. Implement the retrieval component of the IR system in the template provided. Use the TF-IDF vector representation for representing documents.

--

1. Implement the following evaluation measures in the template provided
(i). Precision@k, (ii). Recall@k, (iii). $F_{0.5}$ score@k, (iv). AP@k, and
(v) nDCG@k.

Precision@k:

Recall@k:

$F_{0.5}$ score@k:

AP@k:

nDCG@k:

2. Assume that for a given query, the set of relevant documents is as listed in `incran_qrels.json`. Any document with a relevance score of 1 to 4 is considered as relevant. For each query in the Cranfield dataset, find the Precision, Recall, F-score, average precision, and nDCG scores for $k = 1$ to 10. Average each measure over all queries and plot it as a function of k . The code for plotting is part of the given template. You are expected to use the same. Report the graph with your observations based on it.

Graph:

Observation:

3. Using the `time` module in Python, report the run time of your IR system.

[Warm up] Part 4: Analysis

[Theory]

1. What are the limitations of such a Vector space model? Provide examples from the cranfield dataset that illustrate these shortcomings in your IR system.

Limitations:

Examples from your results:

Part 4: Improving the IR system

Based on the factual record of actual retrieval failures you reported in the assignment, you can develop hypotheses that could address these retrieval failures. You may have to identify the implicit assumptions made by your approach that may have resulted in undesirable results. To realize the improvements, you can use any method(s), including hybrid methods that combine knowledge from linguistic, background, and introspective sources to represent documents. Some examples taught in class are Latent Semantic Analysis (LSA) and Explicit Semantic Analysis (ESA).

You can also explore ways in which a search engine could be improved in aspects such as its efficiency of retrieval, robustness to spelling errors, ability to auto-complete queries, etc.

You are also expected to test these hypotheses rigorously using appropriate hypothesis testing methods. As an outcome of your work, you should be able to make a statement of structure similar to what was presented in the class:

An algorithm A_1 is better than A_2 with respect to the evaluation measure E in task T on a specific domain D under certain assumptions A .

Note that, unlike the assignment, the scope of this component is open-ended and not restricted to the ideas mentioned here. For each method, the final report must include a critical analysis of results; methods can be combined to come up with improvisations. It is advised that such hybrid methods are well founded on principles and not just ad hoc combinations (an example of an ad hoc approach is a simple convex combination of three methods with parameters tuned to give desired improvements).

You could either build on the template code given earlier for the assignment or develop from scratch as demanded by your approach. Note that while you are free to use any datasets to experiment with, the Cranfield dataset will be used for evaluation. The project will be evaluated based on the rigor in

methodology and depth of understanding, in addition to the quality of the report and your performance in Viva.

Your project report (for Part 4) should be well structured and should include the following components.

1. An introduction to the problem setting,
2. The limitations of the basic VSM with appropriate examples from the dataset(s),
3. Your proposed approach(es) to address these issues,
4. A description of the dataset(s) used for experimentations,
5. The results obtained with a comparative study of your approach has improved the IR system, both qualitatively and quantitatively.

The latex template for the final report will be uploaded on Moodle. You are instructed to follow the template strictly.