# CS6370: Natural Language Processing
## Project

Release Date: 24th March 2024 Deadline: 8th May 2025

Name: Roll No.:

| | |
|---|---|
| Amizhthni P R K | EE21B015 |
| Sudarshana K | EE21B142 |
| Aadarsh Kalyaan S | EE21B001 |
| Leshya A | EE21B078 |
| Fardeen Razif | EE21B046 |

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, stopword 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.

[Warm up] Part 1: Working out a toy IR system [Numerical]

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.

Herbivores: {d1}
Carnivores: {d2}
Typically: {d1,d2}
plant:{d1,d2}
meat:{d1,d2}
eaters:{d1,d2}
Deers: {d3}
eat: {d3}
grass: {d3}
leaves: {d3}

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

| | Counts, tfi | | | | Weights, wi = tfi*IDFi | | |
|---|---|---|---|---|---|---|---|
| Terms | d1 | d2 | d3 | IDFi=log(D/dfi) | d1 | d2 | d3 |
| Herbivores | 1 | 0 | 0 | log(3/1)=0.4771 | 0.4771 | 0 | 0 |
| Carnivores | 0 | 1 | 0 | log(3/1)=0.4771 | 0 | 0.4771 | 0 |
| Typically | 1 | 1 | 0 | log(3/2)=0.1761 | 0.1761 | 0.1761 | 0 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| plant | 1 | 1 | 0 | log(3/2)=0.1761 | 0.1761 | 0.1761 | 0 |
| meat | 1 | 1 | 0 | log(3/2)=0.1761 | 0.1761 | 0.1761 | 0 |
| eaters | 2 | 2 | 0 | log(3/2)=0.1761 | 0.3522 | 0.3522 | 0 |
| Deers | 0 | 0 | 1 | log(3/1)=0.4771 | 0 | 0 | 0.4771 |
| eat | 0 | 0 | 1 | log(3/1)=0.4771 | 0 | 0 | 0.4771 |
| grass | 0 | 0 | 1 | log(3/1)=0.4771 | 0 | 0 | 0.4771 |
| leaves | 0 | 0 | 1 | log(3/1)=0.4771 | 0 | 0 | 0.4771 |

where:
$tf_i$ = Term frequency of $i^{th}$ term
$IDF_i$ = Inverse Document Frequency of $i^{th}$ term
$D$ = Total number of documents
$df_i$ = number of documents which has $i^{th}$ term
$w_i = tf_i * \log(D/df_i)$

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

{d1,d2}
Only documents 1 and 2 have words in common with the query (plant and eaters)
This process is detailed in the next problem using cosine similarity

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

**Cosine Similarity calculations:**
Note that for all other words except “Plant” and “Eaters” the count in Q is 0

| Word | tf of Q | IDFi (wrt documents) | tf*(IDFi) of Q |
|---|---|---|---|

| | | | |
|---|---|---|---|
| plant | 1 | 0.1761 | 0.1761 |
| eaters | 1 | 0.1761 | 0.1761 |

As only documents d1 and d2 will be retrieved, d3 is just shown for better illustration.

$$|d1|=\sqrt{0.4771^2 + 0.1761^2 * 3 + 0.3522^2} = 0.667$$
$$|d2|=\sqrt{0.4771^2 + 0.1761^2 * 3 + 0.3522^2} = 0.667$$
$$|d3|=\sqrt{0.4771^2 * 4} = 0.954$$
$$|Q|=\sqrt{0.1761^2 * 2} = 0.249$$

Q.d1 = 0.1761*0.1761 + 0.1761*0.3522 = 0.093
Q.d2 = 0.1761*0.1761 + 0.1761*0.3522 = 0.093
Q.d3 = 0

$$\text{cosine } \theta_{d1} = \frac{0.093}{0.667*0.249} = 0.56$$
$$\text{cosine } \theta_{d2} = \frac{0.093}{0.667*0.249} = 0.56$$
$$\text{cosine } \theta_{d3} = 0$$

Since the cosine similarities of d1 and d2 are equal, both have the same rank1 and d3 has rank2.

**Ranking documents:**

The ranking is as follows: *d1=d2>d3*

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

**No**. This ordering is not desirable. The query given is "Plant Eaters", which imply that *Herbivores* or *deers that eat plants and leaves* should get more priority than *Carnivores.*

However in our cosine similarity ranking, we notice that *d2* which has **Carnivores** is ranked higher than *d3* which has **deers**. This is sufficient to prove that the obtained ranking is **False.**

[Warm up] Part 2: Building an IR system [Implementation]

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

| |
|---|
| The method described in the above section is implemented using Python code. The code is attached in the zip folder. |

[Warm up] Part 3: Evaluating your IR system [Implementation]

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.

Implemented in ***evaluation.py***

**Precision@k:** Proportion of the top k retrieved documents that are relevant. Relevant retrieved documents in top *k* / *k*

**Recall@k:** Proportion of relevant documents that are retrieved in the top k results. Relevant retrieved documents in top *k* / Total relevant documents

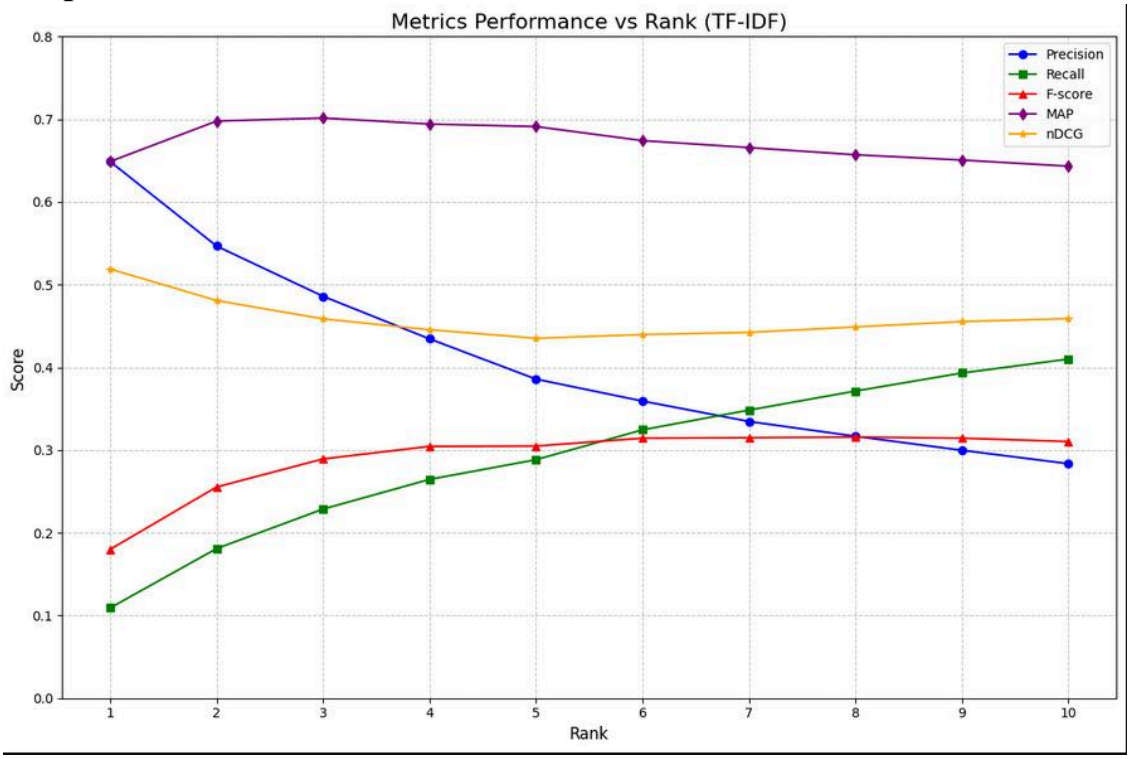
**$F_{0.5}$ score@k:** Harmonic mean of Precision@k and Recall@k, with more weight on precision. $(1 + 0.5^2) \times (\text{Precision} \times \text{Recall}) / (0.5^2 \times \text{Precision} + \text{Recall})$

**AP@k:** Average of precision scores at ranks where relevant documents appear, up to rank *k*. *Rewards early correct retrievals more.*

**nDCG@k:** Measures ranking quality using graded relevance, penalizing relevant items appearing lower in the list.

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:**

1. **Precision**

   Precision decreases with rank as expected, indicating that adding more documents lowers average relevance.

2. **Recall**

   Recall increases with rank as expected, indicating that more relevant documents are retrieved as more retrieved documents are considered.

   The classic opposing trend of Precision and Recall for increasing the number of documents retrieved is seen.

3. **F-score**

   Increases initially, but stagnates due to the opposing trends of

precision and recall.

4. **MAP (Mean Average Precision)**

   Increases initially, but declines and stabilizes showing that the relevant documents are ranked early on.

5. **nDCG (Normalized Discounted Cumulative Gain)**

   Slight decrease, followed by an increase and stabilization showing that relevant documents are retrieved at lower ranks.

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

```
Total time to build IR system from scratch:
26.211757499986 56s
Time to retrieve docs per query after the IR system is built:
0.0042396524445050 29s
```

[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:**
The vector space model assumes independence and orthogonality between all words. This is not a valid assumption. On top of that, the vectors are very large and sparse because of the size of the vocabulary, as a result of which, it takes a considerable amount of time to compute the cosine similarity.
The other shortcomings of this model include:
– Failure to capture polysemy and synonymy causes vocabulary mismatch problems.
– The term document matrix is very large and computing similarity is computationally very intensive. Moreover, addition of a new document requires us to recalculate all the vectors.

– The sequential ordering in which the terms appear in a document is lost in the vector space representation.
– It has limited conceptual understanding and doesn't capture the semantic relationship between words. It just represents documents as a linear combination of words and is purely a lexical matching approach.

**Examples from your results:**

{Query} - Location of halted movement for a non sharp object within a supersonic flow
{doc ID needed} - 35
{doc IDs obtained} - 228 (totally irrelevant), 1300 (bluntness, supersonic are mentioned), 401 (Hypersonic, airflow), 690(Flow, spiked cylinder)

{Query} - Some research on high speed flutter exploration into phenomenon of rapid self excited vibrations of a wing of an aircraft
{doc ID needed} - 1111
{doc IDs obtained} - 100 (vibration, aircraft), 908 (vibration), 51(aircraft), 345(aircraft), 844(vibrations)

{Query} - Constraints on structuring for cost effectiveness in missile frameworks
{doc ID needed} - 834
{doc IDs obtained} - 32(missile), 1217(constraints), 256(totally irrelevant), 834(missile)

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 $\boldsymbol{A_1}$ is better than $\boldsymbol{A_2}$ with respect to the evaluation measure $\boldsymbol{E}$ in task $\boldsymbol{T}$ on a specific domain $\boldsymbol{D}$ under certain assumptions $\boldsymbol{A}$.

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.