CS6370 Assignment 2
Information Retrieval System

Keerthana S
Computer Science Department
Indian Institute of Technology, Madras
Chennai, India
cs20b039@smail.iitm.ac.in

Vaibhav Mahapatra
Mechanical Engineering Department
Indian Institute of Technology, Madras
Chennai, India
me19b197@smail.iitm.ac.in

***Abstract*—In this document, we will be illustrating the approach we've taken to solve questions in Assignment 2 of CS6370: NLP. The assignment focuses on building a search engine from scratch, which is an example of an information retrieval system. This module involves implementing an Information Retrieval system using the Vector Space Model.**

I. Inverted Index Representation

The sentences/documents are
S1 : Herbivores are typically plant eaters and not meat eaters
S2 : Carnivores are typically meat eaters and not plant eaters
S3 : Deers eat grass and leaves

After pre-processing considering {and, are, not} to be stopwords, the documents are
S1 : {'Herbivores', 'typically', 'plant', 'eater', 'meat', 'eater'}
S2 : {'Carnivores', 'typically', 'meat', 'eater', 'plant', 'eater'}
S3 : {'Deers', 'eat', 'grass', 'leaf'}

The code used for preprocessing of the documents is here. Hence the inverted index representation of the documents will be

'Herbivores'	{S1}	$[1,0,0]^T$
'typically'	{S1, S2}	$[1,1,0]^T$
'plant'	{S1, S2}	$[1,1,0]^T$
'eater'	{S1, S2}	$[1,1,0]^T$
'meat'	{S1, S2}	$[1,1,0]^T$
'Carnivores'	{S2}	$[0,1,0]^T$
'Deers'	{S3}	$[0,0,1]^T$
'eat'	{S3}	$[0,0,1]^T$
'grass'	{S3}	$[0,0,1]^T$
'leaf'	{S3}	$[0,0,1]^T$

II. TF-IDF Vector Representations

TF-IDF of a term is given by $TF * IDF$ where TF is the frequency of the term in the document and IDF is its Inverse Document Frequency $\log_2\left(\frac{N}{n}\right)$ where N is the total number of documents in the corpus and n is the number of documents in which the type occurs.
The term frequency matrix : Let the basis vectors be ['Herbivores', 'typically', 'plant', 'eater', 'meat', 'Carnivores', 'Deers', 'eat', 'grass', 'leaf'] and [S1, S2, S3].

$$\begin{bmatrix} 1 & 1 & 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 2 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}^T$$

The IDFs of the terms are :

$$\begin{bmatrix} 1.585 & 0.585 & 0.585 & 0.585 & 0.585 & 1.585 & 1.585 & 1.585 & 1.585 & 1.585 \end{bmatrix}^T$$

The TF-IDF term-document matrix would be :

$$\begin{bmatrix} 1.585 & 0 & 0 \\ 0.585 & 0.585 & 0 \\ 0.585 & 0.585 & 0 \\ 1.170 & 1.170 & 0 \\ 0.585 & 0.585 & 0 \\ 0 & 1.585 & 0 \\ 0 & 0 & 1.585 \\ 0 & 0 & 1.585 \\ 0 & 0 & 1.585 \\ 0 & 0 & 1.585 \end{bmatrix}$$

III. Documents retrieved for a given query

The query 'plant eaters' is preprocessed (code here) to give the terms {'plant, 'eater'}. From the inverted index, the documents for 'plant' are {S1, S2}, the documents for 'eater' are {S1, S2}. Hence the documents retrieved for the given query will be S1 and S2.

IV. Cosine Similarity

The TF-IDF representation of the query is

$$\begin{bmatrix} 0 & 0 & 0.585 & 0.585 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$$

S1: $\begin{bmatrix} 1.585 & 0.585 & 0.585 & 1.170 & 0.585 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$
S2: $\begin{bmatrix} 0 & 0.585 & 0.585 & 1.170 & 0.585 & 1.585 & 0 & 0 & 0 & 0 \end{bmatrix}^T$

Cosine similarity between any vectors $\vec{a}$ and $\vec{b}$ is given by

$$cos\_sim = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|}$$

Cosine similarity between query and S1 = 0.560
Cosine similarity between query and S2 = 0.560
(code for the same is here)

Hence the ranking of the documents would place S1 and S2 on the same level.

V. Is the ranking the best?

In the ranking calculated, documents S1 and S2 have the same cosine similarity with the query, which would mean they have the same rank. However, by intuitive logic we'd like that S1 gets ranked higher than S2, since it talks about herbivores who are plant eaters rather than S2 whose content is about carnivores. The ranking is not the best and can be improved.

VI. Implement an IR System for the Cranfield Dataset

The code for the IR System using the Vector Space Model is attached.

VII.

A. IDF of term that occurs in every document

For a term that occurs in every document, the IDF would be $\log_2\left(\frac{N}{n}\right)$ where $N = n$. Hence,

$$\text{IDF} = \log 1 = 0.$$

The IDF being 0 tells us that there is no discriminating information contained in the term.

B. Is IDF always finite?

When the term under consideration does not occur in any of the documents, IDF becomes $\log_2\left(\frac{N}{n}\right)$ where $n = 0$, which is infinite.
The formula can be modified as $\log_2\left(\frac{N+V}{n+1}\right)$ where V is the total number of types in the dictionary. By doing this, we are counting every type once more than the actual number of times it occurs in the documents, so that none of the types have a count of zero. This will ensure that the IDF never becomes infinite.

VIII. Other Similarity/Distance Measures

Euclidean Distance

The Euclidean Distance [1] measure is the L2 norm between two vectors (say $\vec{A}$ and $\vec{B}$), $||\vec{A} - \vec{B}||_2$. Since queries are usually a lot shorter than documents, their Euclidean distance would give large values even if the query and document were similar. It is not necessary to measure the difference in magnitudes of the query and document vectors, and hence, Cosine Similarity is preferred over Euclidean Distance.

Manhattan Distance

This [2] is the L1 norm between two vectors $||\vec{A} - \vec{B}||_1$. Cosine similarity is preferred over Manhattan Distance for the same reasons as Euclidean distance, the difference in the magnitudes of queries and documents need not be recorded by the distance measure.

Jaccard Similarity

It is defined as the size of the intersection of non-zero dimensions between vectors divided by the size of the union of non-zero dimensions between vectors. Despite being easy to interpret, it doesn't take into account the TF-IDF values (which carry valuable information about the local and global relevance of terms to the documents) that are the dimensional coefficients. Since Cosine Similarity does this, it is preferred.

IX. Why is Accuracy Not an Evaluation Metric?

	Retrieved	Not retrieved
Relevant	A	B
Not Relevant	C	D

Accuracy is given as $\frac{A+D}{A+B+C+D}$. It is not used as an evaluation metric since the value of D (true negatives) could be very large compared to A, B or C when the corpus size is large. Hence accuracy values will always be close to 1 and not sensitive to A, B or C, making it an unsuitable evaluation measure. In such a case where true-negatives are high, we would prefer null-invariant metrics like precision, recall, F-score, etc.

X. When does F_α give more weightage to recall?

The F_α measure is given by

$$\frac{PR}{(1-\alpha)P + (\alpha)R}$$

$$= \frac{1}{\frac{(1-\alpha)}{R} + \frac{\alpha}{P}}, \alpha \in [0, 1]$$

where P = precision, R = recall and α is the weightage given to precision.

When $\alpha = 0$, the F_α measure is purely recall. When $\alpha \in (0, 0.5)$, precision is given more weightage than recall since $1 - \alpha \in (0.5, 1)$.

Hence for the values $[0, 0.5)$ of α, recall is given more weightage than precision.

XI. What shortcoming of Precision @ k is addressed by Average Precision @ k

The Precision @ k (P@k) metric is a measure of how many of the top k documents retrieved are actually relevant. However, this metric does not take into account the order in which the documents are presented to the user. On the other hand, Average Precision @ k (AP@k) takes into account both the relevance of the documents and their rank. It calculates the precision of the relevant documents **up to** a certain rank and averages them, rather than just looking at the precision at that rank.

$$P@k = \frac{N_k}{k}$$

$$AP@k = \frac{\sum_{i=1}^{k} P@i}{N_k}$$

where N_k is the number of relevant documents in k retrieved documents.

Therefore, the main shortcoming of Precision @ k is that it does not consider the order of the retrieved documents, whereas AP @ k addresses this limitation by taking into account both relevance and rank.

XII. Difference between Mean Average Precision (MAP) @ k and Average Precision @ k metrics

Mean Average Precision (MAP) @ k evaluates the Average Precision @ k across multiple queries, and takes their mean. The main difference between MAP @ k and AP @ k is that MAP @ k takes into account the average precision across multiple queries, whereas AP @ k only considers the precision for a single query.

$$MAP@k = \frac{\sum_{i=1}^{N} AP@k}{N}$$

where N is the number of queries. As an evaluation metric, MAP @ k is more comprehensive as it considers the performance of the system across a larger number of queries. It is useful when evaluating the overall performance of an Information Retrieval system, particularly when dealing with large datasets.

XIII. Q13. What's better for the Cranfield Dataset - nDCG or AP?

nDCG is calculated as $\frac{DCG}{iDCG}$ where

$$DCG = \sum_{i=1}^{N} \frac{rel_i}{\log_2(i+1)}$$

where rel_i is the graded relevance of the retrieved document at position i. $iDCG$ is also calculated similarly but with the ideal retrieved documents and the corresponding ideal rel_i values.

nDCG is different from AP@k in that it uses graded relevance instead of a binary 1 if relevant, 0 if not, as in AP@k. Since the Cranfield dataset includes human relevance judgements for the documents apart from binarily stating if it is relevant to the query or no, nDCG is preferred over AP@k.

XIV. Implementation of Evaluation Metrics for the IR System

The code for the five evaluation metrics is attached.

XV. Plotting Evaluation Measures

The plot obtained by averaging the evaluation measures over all queries and plotting them as a function of k is given in Fig. 1.

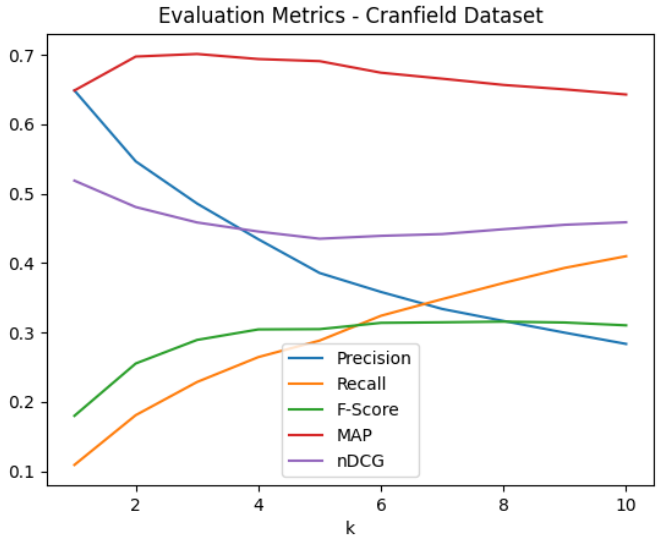

Figure 1. Plot of Evaluation Metrics against k

Observations

- Recall is a monotonically increasing graph with respect to k. This is according to expectation, since the denominator (number of relevant documents) is a constant and the numerator (number of relevant documents that are retrieved) must only increase with rank.
- Precision decreases with rank. This is a good sign since we would desire that more relevant documents be retrieved among the first few ranks than the later ones.
- The F_1 score appears low in general. It increases for the first few values of k and then flattens.
- The nDCG also decreases at first and then stabilizes, showing that the performance of the system when it comes to ranking the relevant documents is constant throughout the ranks.
- MAP starts at the same value as Precision (which we would expect from their formulae) and peaks at a low value of k and then decreases gradually.

XVI. Analysis of Results of Search Engine

- Since the IR system works by finding exact matches of words in queries and documents it is not good at handling
  - Spelling mistakes
  - Alternate spellings
  - Synonymy, Polysemy

  For instance, observe the results retrieved for these queries:

  1) `Enter query below`
     `airfoil shape`
     `Top five document IDs :`
     `194`
     `39`
     `70`

```
1267
467

Enter query below
aerofoil shape
Top five document IDs :
249
206
652
203
676
```

2)
```
Enter query below
Papers on Aerodynamics
Top five document IDs :
925
137
1066
1379
860

Enter query below
Papers on Aero dynamics
Top five document IDs :
22
286
1166
516
948
```

In both 1) and 2), the two queries mean the same thing, but the IR cannot handle alternate spellings (the American airfoil and the British aerofoil) or accidental space insertion, and hence returns different sets of documents for each of the queries. Similarly, it would not be able to identify that the synonyms of the query words might be in the documents, as well.

- Consider the example:

```
Enter query below
dynamics of a dissociating gas
Top five document IDs :
317
110
656
625
167
```

Here, document 110's title is `"dynamics of a dissociating gas"` and document 317's title is `"inviscid hypersonic flow past blunt bodies"`. We would naturally prefer that 110 was returned at a higher rank than 317, but that is not the case in the system. Due to the bag of words model, the IR system ignores the central concept of the document which is evinced by the title and simply matches the words in queries and documents.

- The query words can be presented in any order (sometimes the order might change the meaning) but the same set of documents are retrieved in all cases, regardless of how word order affects the meaning.

XVII. Shortcomings of using a Vector Space Model for IR [3]

The Vector Space Model being a Bag of Words model, does not take into account word meaning or word order. Like the ones shown in the previous answer, there are some shortcomings to this model :

- The model makes the assumption that the representations of the words are orthogonal to every other word in the corpus. This is not be true since synonyms and polysemous words exist, meaning that there might be overlap between the concepts denoted by various words and they are not necessarily orthogonal.
- We have used only unigrams in the inverted index, and the Vector Space information retrieval also does not take word order into account. This is not optimal for a search engine, since without word order we cannot model the contexts of terms as well.
- When we begin expanding the search engine to larger corpuses and larger dictionaries, the representation of each document becomes very large and hence the system becomes computationally expensive.
- When it is required to expand the model to more documents, adding even a single word to the vocabulary requires recalculation of the representations of every document in the space.
- The model does not account for spelling mistakes in the documents or the queries. Given that the document sizes are small, spell-checked words might be vital.

XVIII. Including the Title in Document Representation

In the IR Vector Space Model, the documents are represented using TF-IDF values. One way to include the title and weigh it thrice as much as the body of the document would be to triple the TF-IDF values of the title words and include them in the representation of the document. One way of looking at it would be to imagine that the document contains three copies of the title along with its body.

Thrice might be a good number for the Cranfield dataset since the documents are small in size. For datasets with larger, or more variable document lengths, the number of times we want to weigh the title might be a hyperparameter we have to optimise, since we wouldn't want to ignore the title or overemphasize on it.

XIX. Using Bigrams instead of Unigrams in the Index

Advantages

- Word order can be modeled to some extent by using bigrams to index the documents, this would include the

context in which the words are used, along with the words themselves.
- This would improve precision, since most of the retrieved documents would be the ones containing the exact two-word phrases as in the query, making them more likely to be relevant to the query.

Disadvantages

- Usage of bigrams over unigrams would make the system much more computationally expensive. The index would have to be very large compared to the unigram case.
- The index will be created using bigrams in the corpus. The bigrams in the query will have to be looked up in the index. There is a chance that the bigrams in the query are not present in the corpus (despite being similar in meaning to bigrams in the corpus), so they will have to be ignored since they don't map to any documents in the index, and their information content will be lost.
- Without taking word sense into account, we will be losing out on recall. Documents which use the same two-word phrase as the query but with a synonym substituted for a word will not be retrieved despite meaning the exact same thing as the two-word phrase in the queries. Some highly relevant documents might be missed out on.

XX. Relevance Feedback from the User

- When the search engine presents the retrieved documents to the user, the ones on which the user clicks can be given a higher relevance score for that query, than the ones ignored by the user. [4]
- The time that the user spends on a retrieved document can be tracked. The more the time spent on a retrieved document, the greater its relevance score.
- Such data can be collected from several users and how they engage with the system. Results that have higher page visits and longer time spent across all users can be judged as more relevant than other results.

References

[1] 3 basic distance measurement in text mining. https://towardsdatascience.com/3-basic-distance-measurement-in-text-mining-5852becff1d7.

[2] Taxicab geometry. https://en.wikipedia.org/wiki/Taxicab_geometry.

[3] Vector space model. https://en.wikipedia.org/wiki/Vector_space_model.

[4] Fetching relevant information in the background. https://people.ischool.berkeley.edu/~hearst/irbook/10/node8.html.