Assignment 2

CS6370 - Natural Language Processing

Amogh Prabhunanda Patil - EE19B134
Abhigyan Chattopadhyay - EE19B146

March 22, 2022

1 Vector Space Model

1. Now that the Cranfield documents are pre-processed, our search engine needs a data structure to facilitate the 'matching' process of a query to its relevant documents. Let's work out a simple example. Consider the following three sentences:

1. **S1 Herbivores are typically plant eaters and not meat eaters**
2. **S2 Carnivores are typically meat eaters and not plant eaters**
3. **S3 Deers eat grass and leaves**

Assuming are, and, not as stop words, arrive at an inverted index representation for the above documents (treat each sentence as a separate document)

An inverted index is an index data structure storing a mapping from content, such as words to a set of documents.

The following is an inverted index representation of the above documents:

Herbivores	[1, 0, 0]
typically	[1, 1, 0]
plant	[1, 1, 1]
eaters	[1, 1, 0]
meat	[1, 1, 0]
Carnivores	[0, 1, 0]
Deers	[0, 0, 1]
eat	[0, 0, 1]
grass	[0, 0, 1]
leaves	[0, 0, 1]

2. Next, we must proceed on to finding a representation for the text documents. In the class, we saw about the TF-IDF measure. What would be the TF-IDF vector representations for the documents in the above table? State the formula used.

TF-IDF value is computed for a given word of the dictionary with respect to the given document.

$$\text{IDF} = \text{TF} \cdot \log\left(\frac{N}{df}\right) \tag{1}$$

Here,

1. TF = Frequency of word in the given document,
2. N = Total number of documents,
3. *df* - The number of documents in which the word occurs in.

The basis vector is as follows:

```
['Herbivores', 'typically', 'plant', 'eaters', 'meat', 'Carnivores',
'Deers', 'eat', 'grass', 'leaves']
```

The TF-IDF representation of the Documents in the above basis vectors:

```
D1 - [0.477,    0.176,  0.176,  0.352,  0.176,  0., 0., 0., 0., 0.]
D2 - [0.,   0.176,  0.176,  0.352,  0.176,  0.477,  0., 0., 0., 0.]
D3 - [0.,   0.,  0.,    0., 0., 0., 0.477,  0.477,  0.477,  0.477]
```

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

- The documents retrieved for the word : 'plant'
  - Documents - D1, D2
- The documents retrieved for the word : 'eaters'
  - Documents - D1, D2
- The documents retrieved for the word : 'plant eaters'
  - Documents - D1, D2

4. Find the cosine similarity between the query and each of the retrieved documents. Rank them in descending order.

- Query in the TF-IDF representation:

```
Q1 - [0.,   0., 0.176,  0.176,  0., 0., 0., 0., 0., 0.]
```

- Cosine similarity with the documents :

```
Q1, D1 - 0.560156917515788
Q1, D2 - 0.560156917515788
Q1, D3 - 0.0
```

- Rank of the retrieved documents:

```
1 - D1
2 - D2
3 - D3
```

5. Is the ranking given above the best?

The ranking is not the best.
The third document which talks about deer (a plant eater) and the fact that deer eat grass enforcing the plant-eating nature of the deer, makes it more relevant than the second document which primarily talks about carnivores.

The best ranking would be:

1. D1
2. D3
3. D2

6. Now, you are set to build a real-world retrieval system. Implement an Information Retrieval System for the Cranfield Dataset using the Vector Space Model.

The code for the same is attached in the submission.

7. Correcting IDFs for terms

1. What is the IDF of a term that occurs in every document?

2. Is the IDF of a term always finite? If not, how can the formula for IDF be modified to make it finite?

1. The IDF would be:
$$\text{IDF} = \log\left(\frac{N}{N}\right) = \log(1) = 0 \tag{2}$$

2. If a query word is not available in the data set dictionary its IDF would be:
$$\text{IDF} = \log\left(\frac{N}{0}\right) = \infty \tag{3}$$

   We modify the formula as follows to avoid this issue:
$$\text{IDF} = \text{TF} \cdot \log\left(\frac{N}{df+1}\right) \tag{4}$$

   Here,

   (a) TF = Frequency of word in the given document,

   (b) N = Total number of documents,

   (c) *df* - The number of documents in which the word occurs in.

8. Can you think of any other similarity/distance measure that can be used to compare vectors other than cosine similarity. Justify why it is a better or worse choice than cosine similarity for IR.

1. **Euclidean Distance Measure:**
$$\text{Distance} = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2} \tag{5}$$

   - Lower the distance between two vectors, more is the similarity between them.
   - However we can see that as the dimension of the vectors increase, the number of terms in the summation will increase, resulting in a higher distance measure.
   - Cosine Similarity is a better choice as it is scale-invariant.

2. **Manhattan Distance:**
$$\text{Distance} = \sum_{i=1}^{n} |q_i - p_i| \tag{6}$$

   - It can be seen that as the dimension of the vectors increase, the number of terms in the summation will increase, resulting in a higher distance measure.
   - Cosine Similarity is a better choice as it is scale-invariant.

3. **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. Higher ratio implies higher level of similarity.
   - This distance measure disregards the values of the dimensional coefficients (TF-IDF values of words). Therefore Cosine Similarity performs better.

9. Why is accuracy not used as a metric to evaluate information retrieval systems?

Accuracy is essentially the fraction of classifications that are correct.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{7}$$

Here,

1. TP = True Positives
2. TN = True Negatives
3. FP = False Positives
4. FN = False Negatives

There are two actual classes, relevant and non-relevant, and an information retrieval system can be thought of as a two-class classifier.

The data is extremely skewed: normally over 99.9% of the documents are in the non-relevant category.

A system tuned to maximize accuracy can appear to perform well by simply marking all documents as irrelevant to all queries (basically not retrieving anything will be given a high score).

Since we are not returning any positives and the number of false negatives are very low (very few actually relevant documents exist in a large corpus), the following accuracy metric (which will arise when we don't return any positives) will be very high:

$$\text{Accuracy} = \frac{\text{TN}}{\text{TN+FN}} \tag{8}$$

As a result, using accuracy as a metric to evaluate IR systems will reward it for producing no results and hence it is not used.

10. For what values of α does the F_α-measure give more weightage to recall than to precision?

In the F_α measure, the recall is considered to be $\frac{1-\alpha}{\alpha}$ times as important as the precision, i.e.:

$$F_\alpha = \frac{1}{\frac{\alpha}{P} + \frac{1-\alpha}{R}} \tag{9}$$

where P = precision and R = recall.

α can be varied from 0 to 1, both inclusive.

Hence, taking a value of 0.5, they both are given equal importance, taking values between 0 and 0.5 gives more importance to recall and values between 0.5 and 1 gives more importance to precision.

At α exactly 0, $F_\alpha = R$, and at α exactly 1, $F_\alpha = P$, which are highly skewed in one direction or the other.

An ideal value of α would be somewhere between 0.25 and 0.75, depending on which of P or R we want to give a higher relevance.

11. What is a shortcoming of Precision @ k metric that is addressed by Average Precision @ k?

Precision @ k essentially counts how many relevant results are present in the top k retrieved documents.

The main issue using this metric is that it does not rank the order of relevance of the documents in the top k.

[1,1,1,1,1,0,0,0,0,0] and [0,0,0,0,0,1,1,1,1,1]

Given the above two relevance lists, both have the same precision score, even though the first list ranks the documents better than the second one.

Average precision @ k mitigates this issue by taking the weighted sum of precisions with the binary relevance metric divided by the total number of relevant documents:

$$\text{Average Precision @ k} = \frac{\sum_{j=1}^{k} P_j \cdot \text{rel}_j}{N} \tag{10}$$

where N is the number of relevant documents, and rel_j is 1 if the document is relevant else 0.

12. What is Mean Average Precision (MAP) @ k? How is it different from Average Precision (AP) @ k?

The Mean Average Precision @ k (mAP_k) over N queries is defined as :

$$mAP_k = \frac{\sum_{i=1}^{N} AP_{k_i}}{N} \tag{11}$$

Mean average precision for a set of queries is the mean of the average precision scores for each query. Unlike AP_k which is defined for a single query, mAP_k is defined for a set of queries, therefore better indicative of the performance of an IR system.

13. For Cranfield dataset, which of the following two evaluation measures is more appropriate and why? (a) AP (b) nDCG

The Cranfield dataset is a small dataset that has extremely specific documents and queries, unlike most queries used by users. As a result, it skews our information retrieval to prefer nDCG over AP.

The Cranfield dataset has a rank that shows which document is ranked higher besides just mentioning whether it is relevant to a query. nDCG is able to handle a non-binary relevance metric as we have here, unlike AP which is specifically only able to handle binary relevance data.

Hence, the nDCG metric is a much better metric to handle the Cranfield dataset with such detailed rank information provided to us.

14. Implement the following evaluation metrics for the IR system:

- **Precision @ k**
- **Recall @ k**
- **F-Score @ k**
- **Average Precision @ k**
- **nDCG @ k**

The code for the same is attached in the submission.

15. Assume that for a given query, the set of relevant documents is as listed in cran_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 function of k. 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.

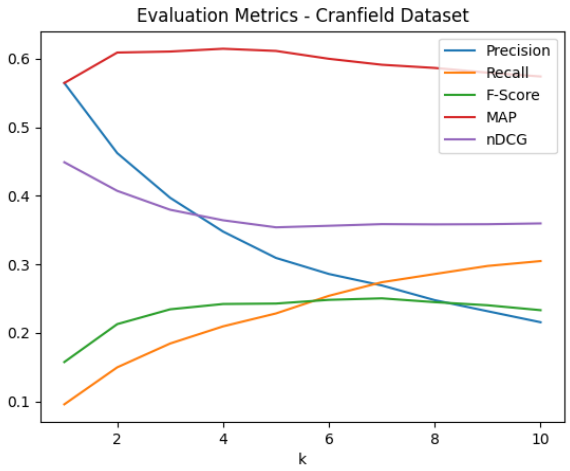

Figure 1

- Precision decreases monotonically with k. This is as expected for a good IR system as the density of relevant documents is expected to decrease with increase in retrieved documents(as K increases).
- Recall monotonically increases with k as expected. Since total number of relevant documents is a constant (the denominator is constant) and the number of relevant documents retrieved increases with k.
- F-score initially increases with k and then stabilizes. F-score is generally used to compare various models to break the precision-recall trade-of
- Average Precision is also expected to increase monotonically with k from definition, as it is observed from the plot.

- The nDCG value decreases for small K and remains relatively constant for higher K(slight increase).

16. Analyse the results of your search engine. Are there some queries for which the search engine's performance is not as expected? Report your observations.

```
Query : "papers on flow visualization on slender conical wings ."
Top five document IDs :
602
222
19
420
465

Query : "aero"
Top five document IDs :
22
1
2
3
4
```

The Second Query's result 4 does not contain the word aero at all, thus showing a bad example.

17. Do you find any shortcoming(s) in using a Vector Space Model for IR? If yes, report them.

Shortcomings of Vector Space Model:

- The model implicitly assumes the terms are orthogonal, which is often not the case.
- Similar documents having different vocabularies will have a small similarity measure. Two synonyms will be treated as completely different words.
- When a new term is to be included, the entire vector set needs to be recomputed.
- The data terms are assumed to be statistically independent. Context of words play no roles.
- The model is not capable of modelling the sequence of terms.
- With a large dictionary of words, the vector representation of documents becomes large resulting in very calculation intensive operations.

18. While working with the Cranfield dataset, we ignored the titles of the documents. But, titles can sometimes be extremely informative in information retrieval, sometimes even more than the body. State a way to include the title while representing the document as a vector. What if we want to weigh the contribution of the title three times that of the document?

The contribution of the title can be included by adding the TF-IDF representation of the title to the TF-IDF representation of the document. We can triple the TF-IDF values of the title to

weigh the contribution of the title three time that of the document.

In the Cranfield Dataset, the size of the documents aren't extremely large allowing the title to have some effect. For larger documents we would have to weigh the titles higher. However weighing it extremely high, might put over-emphasis on the title. Therefore it is important to strike a good balance in the weighing of the title which is an extremely challenging task given variable length documents.

19. Suppose we use bigrams instead of unigrams to index the documents, what would be its advantage(s) and/or disadvantage(s)?

- Adavantages:
  - Bigrams allow for modelling sequence of terms in documents.
  - Context modelled this way can help in word sense Disambiguation which increases precision of the information retrieval system.
- Disadvantages :
  - The dimensions of the vector space model would be formed from the bigrams found in the corpus. However it is not very unlikely that a bigram in the query is not present in the corpus.
  - From the above point, we can say the Recall will be lesser than the unigram model. However the Bigram model can be modified to include the unigram model under it. But this would significantly increase the dimension of the vector space, increasing space and time overheads.

20. In the Cranfield dataset, we have relevance judgements given by the domain experts. In the absence of such relevance judgements, can you think of a way in which we can get relevance feedback from the user himself/herself? Ideally, we would like to keep the feedback process to be non-intrusive to the user. Hence, think of an 'implicit' way of recording feedback from the users.

1. User Interaction
   When search results are provided and the user happens to click on the higher ranked retrieved documents, the system can take it as implicit positive feedback. Similarly when the user clicks on a lower ranked retrieved document or doesn't click on any, the system can take it as negative feedback. The system can also take the duration of interaction of the user on a retrieved document as feedback. If the user immediately clicks away, it could mean the retrieved document isn't what the user was looking for(not relevant).

2. User Experience
   Enquire if the user is enjoying the search Engine. If the answer is a 'yes', Its more likely that the search engine is providing relevant documents to the queries.

3. Eye Tracking
   Observation of features such as eye fixation and pupil dilation as the user observes the results. The hypothesis is that features such as duration of fixation and diameter of the pupil vary in a systematic way between relevant and non-relevant results; for example, larger pupil diameter might be an indication of relevance.