

<https://github.com/sergigf03/IRWA-2024>

Ranking:

Project Part 3 IRWA

1. Introduction

In this project, our goal is to rank documents based on how relevant they are to a specific query (search term). We will rank the documents using different scoring methods and text representations.

2. Questions and Answers:

a) What two ranking methods should we use and how do they work?

TF-IDF + Cosine Similarity: A classical, common way of obtaining the weights of words in a document is by contrasting these words to a query. The weights of words by their common occurrences in documents and a query represent the relevance of words.

```
Top 20 Ranked Documents by TF-IDF + Cosine Similarity:
Document ID: doc_1273, Cosine Similarity Score: 0.9321757179559246
Document ID: doc_11265, Cosine Similarity Score: 0.9321757179559246
Document ID: doc_16057, Cosine Similarity Score: 0.9321757179559246
Document ID: doc_28464, Cosine Similarity Score: 0.9321757179559246
Document ID: doc_20704, Cosine Similarity Score: 0.9262099832481728
Document ID: doc_17156, Cosine Similarity Score: 0.8883695631933285
Document ID: doc_21006, Cosine Similarity Score: 0.8883695631933285
Document ID: doc_36129, Cosine Similarity Score: 0.8703493953287428
Document ID: doc_18896, Cosine Similarity Score: 0.8653753366684499
Document ID: doc_10548, Cosine Similarity Score: 0.8525353378102423
Document ID: doc_13939, Cosine Similarity Score: 0.8525353378102423
Document ID: doc_35711, Cosine Similarity Score: 0.8525353378102423
Document ID: doc_3210, Cosine Similarity Score: 0.8474567278511329
Document ID: doc_17899, Cosine Similarity Score: 0.8474567278511329
Document ID: doc_28724, Cosine Similarity Score: 0.8474567278511329
Document ID: doc_40576, Cosine Similarity Score: 0.8474567278511329
Document ID: doc_16494, Cosine Similarity Score: 0.8408901649455093
Document ID: doc_20585, Cosine Similarity Score: 0.8408901649455093
Document ID: doc_20631, Cosine Similarity Score: 0.8408901649455093
Document ID: doc_20658, Cosine Similarity Score: 0.8408901649455093
```

This image shows the ranking of documents using **TF-IDF + Cosine Similarity**, where documents are ranked based on their cosine similarity score to the query.

BM25: BM25 is a more sophisticated algorithm, modeling term frequency within documents. It takes into account the different sizes of the documents.

<https://github.com/sergigf03/IRWA-2024>

```
Top 20 Ranked Documents by BM25:
Document ID: doc_17156, BM25 Score: 13.636151512312594
Document ID: doc_30422, BM25 Score: 13.168913200225282
Document ID: doc_37712, BM25 Score: 10.871392391317688
Document ID: doc_10923, BM25 Score: 10.79965400759432
Document ID: doc_21006, BM25 Score: 10.359179922767837
Document ID: doc_20704, BM25 Score: 10.358118906529452
Document ID: doc_37241, BM25 Score: 10.333603630528476
Document ID: doc_46909, BM25 Score: 10.328318411051908
Document ID: doc_37560, BM25 Score: 10.177080587394158
Document ID: doc_37209, BM25 Score: 10.169551073889046
Document ID: doc_37219, BM25 Score: 10.169551073889046
Document ID: doc_47837, BM25 Score: 10.088309541241681
Document ID: doc_47839, BM25 Score: 10.088309541241681
Document ID: doc_47841, BM25 Score: 10.088309541241681
Document ID: doc_47843, BM25 Score: 10.088309541241681
Document ID: doc_47845, BM25 Score: 10.088309541241681
Document ID: doc_47848, BM25 Score: 10.088309541241681
Document ID: doc_48027, BM25 Score: 10.088309541241681
Document ID: doc_48059, BM25 Score: 10.088309541241681
Document ID: doc_48070, BM25 Score: 10.088309541241681
```

It shows the **BM25** ranking results, where documents are ranked by their BM25 score, adjusting for term frequency and document length.

Let's check the difference between these two methods described previously:

	PRO	CONS
TF-IDF	Simple and easy to implement.	Doesn't account for document length or context.
BM25	Accounts for document length and adjusts term frequency, better for various document sizes.	More complex to implement and may require tuning.

Regarding Our Own Score:

For our **Your-Score (P-score)**, we created a formula that combines both social media popularity and textual relevance. We use metrics such as the number of **likes** and **retweets** to determine the importance of a document. We chose this approach because social media popularity can be a strong indicator of relevance, where highly discussed or liked posts are often more important or interesting to the public.

<https://github.com/sergigf03/IRWA-2024>

```
Top-ranked documents by P-Score:
Document ID: doc_3203, P-Score: 47349.600000000006
Document ID: doc_46206, P-Score: 28520.100000000002
Document ID: doc_45142, P-Score: 22561.300000000003
Document ID: doc_38012, P-Score: 20259.5
Document ID: doc_38410, P-Score: 20077.6
Document ID: doc_38262, P-Score: 19929.300000000003
Document ID: doc_35993, P-Score: 17205.0
Document ID: doc_27071, P-Score: 15231.2
Document ID: doc_9846, P-Score: 13910.6
Document ID: doc_38379, P-Score: 12391.0
Document ID: doc_41472, P-Score: 11900.8
Document ID: doc_28154, P-Score: 11785.5
Document ID: doc_18306, P-Score: 11322.7
Document ID: doc_23286, P-Score: 10439.8
Document ID: doc_27974, P-Score: 9859.5
Document ID: doc_31312, P-Score: 9262.6
Document ID: doc_46278, P-Score: 9015.8
Document ID: doc_13132, P-Score: 8902.400000000001
Document ID: doc_28056, P-Score: 8575.8
Document ID: doc_16447, P-Score: 8402.5
```

This capture displays documents ranked by **P-Score**, incorporating social media metrics like likes and retweets to reflect real-world importance.

Let's analyze the advantages and disadvantages of this method used:

PRO	CONS
Social media metrics are incorporated, giving rankings a real-world importance.	Can over-rank popular content that isn't necessarily relevant to the query.
They provide a more coherently complete ranking, infused by both the content aspect as well as the social one.	Popularity metrics may be manipulated, leading to inaccurate rankings.
Helps rank documents that may not have high textual relevance but are highly popular on social platforms.	Requires real-time social media data, which can be challenging to manage and might introduce noise.

<https://github.com/sergigf03/IRWA-2024>

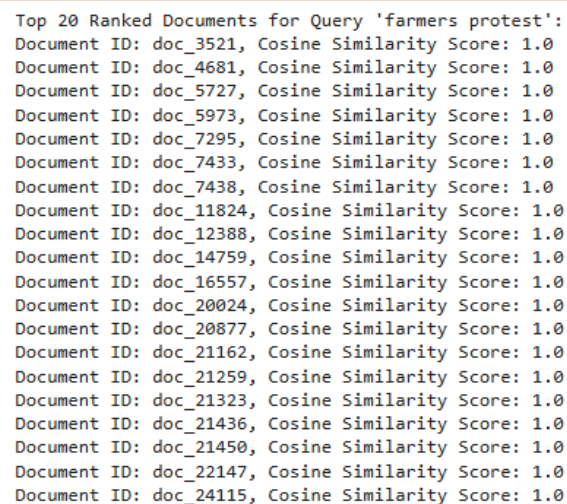
In conclusion, we chose **TF-IDF** for its simplicity and effectiveness with smaller datasets, though it's less suited for longer texts. Moreover, we picked **BM25** because it adjusts for document length, offering a more balanced ranking for longer documents. And finally, **Our-Score** has been added to include social media metrics like tweet popularity, which help rank content based on its real-world importance.

b) How are tweets represented using word2vec?

The whole idea behind this project was to come up with a numerical representation of tweets using **word2vec**. Each word in a tweet is treated as a vector in some high-dimensional space. These vectors of all the words are then averaged to create a single vector that represents the whole tweet. This helps our system understand the meaning of the tweet by considering the relationships between the words.

Ranking with Cosine Similarity

After representing tweets using word2vec, we apply **Cosine Similarity** to measure how closely each document (tweet) matches the query. The cosine similarity score indicates how similar the document is to the query based on the word vectors.



```
Top 20 Ranked Documents for Query 'farmers protest':
Document ID: doc_3521, Cosine Similarity Score: 1.0
Document ID: doc_4681, Cosine Similarity Score: 1.0
Document ID: doc_5727, Cosine Similarity Score: 1.0
Document ID: doc_5973, Cosine Similarity Score: 1.0
Document ID: doc_7295, Cosine Similarity Score: 1.0
Document ID: doc_7433, Cosine Similarity Score: 1.0
Document ID: doc_7438, Cosine Similarity Score: 1.0
Document ID: doc_11824, Cosine Similarity Score: 1.0
Document ID: doc_12388, Cosine Similarity Score: 1.0
Document ID: doc_14759, Cosine Similarity Score: 1.0
Document ID: doc_16557, Cosine Similarity Score: 1.0
Document ID: doc_20024, Cosine Similarity Score: 1.0
Document ID: doc_20877, Cosine Similarity Score: 1.0
Document ID: doc_21162, Cosine Similarity Score: 1.0
Document ID: doc_21259, Cosine Similarity Score: 1.0
Document ID: doc_21323, Cosine Similarity Score: 1.0
Document ID: doc_21436, Cosine Similarity Score: 1.0
Document ID: doc_21450, Cosine Similarity Score: 1.0
Document ID: doc_22147, Cosine Similarity Score: 1.0
Document ID: doc_24115, Cosine Similarity Score: 1.0
```

This screenshot shows the Top 20 Ranked Documents for the query "**farmers protest**" using Cosine Similarity, with all documents having a score of 1.0, indicating a perfect match with the query. We have used Word2vec for 4 other queries, as seen in the ipynb file.

c) What better representation than word2vec could be used?

<https://github.com/sergigf03/IRWA-2024>

Doc2Vec: Unlike word2vec, which focuses on individual words, we use **Doc2Vec** to create vectors for entire documents. This is useful when we want a better overall representation of the text.

Sentence2Vec: Similar to Doc2Vec, but this method focuses on entire sentences. We might use it when tweets are short and don't require a full document representation.

d) Pros and cons:

Method	PROS	CONS
Doc2Vec	It provides a more effective way for longer documents to catch up with the full context.	More complex to train and requires more data and computational resources compared to simpler models like word2vec.
Sentence2Vec	Effective on short texts, such as tweets.	Maybe not effective for longer documents, because it doesn't keep in mind the rest of the whole text.