Albert Jané Lardiés - U215114
Marc de los Aires Tello - U198732
Jordi Esteve Claramunt - U215108
https://github.com/jordiestevee/irwa-search-engine-g_019
TAG: IRWA-2025-part-2

IRWA Project – Part 2: Indexing and Evaluation

Introduction

The purpose of IRWA Part 2 is to build a TF-IDF-based retrieval system capable of indexing and ranking textual data. The project involves implementing an inverted index, computing document relevance using TF-IDF weighting, and evaluating the retrieval performance with standard Information Retrieval (IR) metrics. This part builds upon previous work by focusing on implementing indexing and evaluation functionalities within the retrieval pipeline.

Data and Pre-processing

The dataset used in this project consists of fashion product descriptions. Each entry includes information such as product name, category, material, and gender category. To prepare the data for indexing, several preprocessing steps were performed:

• Tokenization – splitting text into individual tokens (words).

• Stopword removal – eliminating common non-informative words (e.g., 'the', 'and', 'is').

• Stemming – reducing words to their root forms (e.g., 'running' → 'run').

• Lowercasing – converting all tokens to lowercase to ensure uniformity.

Indexing

Index Construction

The system first processes all product descriptions to build an inverted index, a data structure that maps each unique word to the list of documents (products) that contain it. It also calculates how important each term is across the collection using TF-IDF (Term Frequency–Inverse Document Frequency).

Document Representation

Each document is represented as a numerical vector of TF-IDF weights.
The code also precomputes the length (norm) of each document vector to make later similarity calculations faster.

Query Processing

When the user enters a search query, it goes through the same preprocessing pipeline (tokenization, stop-word removal, and stemming) to match the document representation.

Search and Ranking

- The system retrieves only the documents that contain *all* the query terms.
- It then computes a cosine similarity score between the query vector and each candidate document.

Result

Finally, the top-ranked results are displayed with their product information (title, brand, category, and link), ordered by similarity score.

Query Design

To evaluate the search engine's performance, a small set of representative queries was created to reflect typical user search behavior. The aim was to test how effectively the TF-IDF retrieval system ranks relevant products for different types of search intents.

1. **Selection of Common Terms**

   The dataset was first analyzed to identify the most frequent and meaningful words appearing in product descriptions. Extremely common or uninformative terms (such as stopwords or words like "size" and "color") were removed to focus on distinctive vocabulary that captures product characteristics such as gender, material, or style.

```
from collections import Counter
from random import sample

# build frequent token list (already preprocessed tokens/stems)
top_terms = Counter(" ".join(products_df[TEXT_COL].fillna("")).split()).most_common(200)

# filter: remove stopwords, very short tokens and a few generic corpus tokens
filtered = [t for t,_ in top_terms if t not in stop_words and len(t) > 3 and t not in {"size","color"}]

print(filtered[:20])  # show top 20
```

✓ 0.3s

```
['fabric', 'neck', 'sleev', 'type', 'cotton', 'pack', 'style', 'wear', 'round', 'women', 'regular', 'pattern', 'wash', 'code', 'care', 'print', 'shirt', 'solid', 'brand', 'suitabl']
```

2. **Design of Test Queries**

   Based on the most relevant terms, five realistic search queries were defined:

   - *"women cotton shirt"*
   - *"round neck shirt"*
   - *"solid pattern wear"*
   - *"men regular fit"*
   - *"printed cotton fabric"*

3. These queries were chosen to cover different semantic aspects of product search—such as gender, fabric type, style, and pattern—allowing for a balanced evaluation of the retrieval system.

4. **Evaluation Process**
   Each query was submitted to the search engine, which retrieved and ranked the most relevant products using TF-IDF cosine similarity. The top-5 results were inspected to determine whether the system correctly prioritized products that match the user's intent.

```
from collections import Counter
from random import sample

# build frequent token list (already preprocessed tokens/stems)
top_terms = Counter(" ".join(products_df[TEXT_COL].fillna("")).split()).most_common(200)

# filter: remove stopwords, very short tokens and a few generic corpus tokens
filtered = [t for t,_ in top_terms if t not in stop_words and len(t) > 3 and t not in {"size","color"}]

print(filtered[:20])  # show top 20
✓ 0.3s
['fabric', 'neck', 'sleev', 'type', 'cotton', 'pack', 'style', 'wear', 'round', 'women', 'regular', 'pattern', 'wash', 'code', 'care', 'print', 'shirt', 'solid', 'brand', 'suitabl']
```

Evaluation

To assess retrieval performance, several evaluation metrics were implemented, including Precision@K, Recall@K, AP@K, F1@K, Mean Average Precision (MAP), Mean Reciprocal Rank (MRR), and Normalized Discounted Cumulative Gain (NDCG). These metrics capture both precision-oriented and rank-sensitive aspects of the system's performance.

Q1: "women full sleeve sweatshirt cotton"

Q2: "men slim jeans blue"

Table 1: Evaluation Results

Query	Precision@K	Recall@K	AP@K	F1@K	MAP	MRR	NDCG
Q1	0.700	0.538	0.508	0.609	0.703	0.490	1.000
Q2	0.600	0.600	0.473	0.600	0.665		

The retrieval system demonstrates strong performance across the evaluated queries. For the query "women full sleeve sweatshirt cotton," the high precision at K indicates that most of the top-ranked results are relevant, while a moderate recall suggests that some relevant items are missed. Average precision and F1 scores show a balanced retrieval effectiveness, and the mean average precision confirms strong overall performance.

For "men slim jeans blue," both precision and recall are balanced, reflecting consistent retrieval of relevant items, though slightly lower average precision indicates that some top-ranked results may be less relevant. The mean average precision remains solid, though marginally below that of the first query.

Rank-sensitive metrics provide additional insights into the ordering of results. The mean reciprocal rank shows that the first relevant document generally appears near the top of the results, while the normalized discounted cumulative gain indicates excellent rank ordering, with relevant documents effectively prioritized.

Relevance Judgement and Evaluation

This section evaluates the effectiveness of the retrieval system in returning relevant results for user queries. The evaluation involves three main stages: defining ground truth, assessing performance through standard metrics, and analyzing the system's strengths and limitations.

1. Ground Truth Definition

We were asked to asign binary relevance labels to documents for each of the test queries defined earlier and evaluate them.

2. Evaluation Metrics and Interpretation

Several standard IR metrics were used to evaluate the system from different perspectives: precision, completeness, ranking order, and early relevance. Some queries did not have any result with enough relevance, so all metrics were 0. For the rest of the queries, precision was low and recall high, while the rest of metrics varied.

3. **System Analysis and Limitations**

The current retrieval system is based on TF-IDF weighting and cosine similarity, relying mainly on exact keyword matching. While effective for basic searches, several limitations were identified.

a. Retrieval Accuracy

- Issue: Some relevant products are missing or ranked low, especially for multi-attribute queries (e.g., "women cotton shirt").

- Proposed Solutions: Integrate semantic embeddings (e.g., Word2Vec) to capture contextual meaning beyond exact matches.

b. Ranking Quality

- Issue: Relevant results are not always at the top (low MRR).
  Proposed Solutions: Use learning-to-rank algorithms that combine textual and behavioral features (e.g., click data). Implement query expansion or relevance feedback to refine ambiguous queries.

d. Indexing Strategy

- Issue: Current indexing lacks semantic awareness and cannot capture relationships between co-occurring terms.

- Proposed Solutions:

    - Build vector-based indices (for example, using Doc2Vec).

AI Usage Statement

ChatGPT (GPT-5) was used to structure and format this report. All code and results were implemented and verified by the authors.