Project Documentation

Introduction

This document provides an overview and documentation for the TF-IDF based search engine project developed as a final project for the Data Structure course at Iran University of Science and Technology, autumn 2023.

Project Structure

Ensure that you have created a dataset folder and placed the data folder and data.json in it.

Installation

To set up the project on an Ubuntu operating system, follow these steps:
bash
Copy code
sudo apt install python3-venv
python3 -m venv env
source env/bin/activate
pip install -r requirements.txt
python3 main.py

Note: If you install new dependencies, freeze them into requirements.txt for future use:
bash
Copy code
pip3 freeze > requirements.txt

Project Components

tf_idf.py

- This script contains the TFIDFVectorizer class, which includes methods for calculating term frequency, inverse document frequency, and TF-IDF values.
- The class provides functionality for fitting the model to a dataset (fit), transforming documents into TF-IDF representations (transform), and both fitting and transforming in a single step (fit_transform).
- The script includes detailed comments explaining each method's purpose.

dataset_generator.py

- This script contains the DatasetGenerator class responsible for preparing and cleaning the dataset.

- The class includes methods for loading and saving JSON data, converting sets to lists, and generating a clean dataset.
- The script uses the TF-IDF vectorizer from tf_idf.py to calculate TF-IDF values for documents and queries.
- Additionally, it provides methods for calculating TF-IDF values, cleaning and saving the dataset, and loading the dataset from saved files.

helper.py

- This script includes helper functions used throughout the project.
- The cosine_similarity function calculates the cosine similarity between two vectors.
- The search function performs a search using TF-IDF and cosine similarity, returning the most similar document and sentence.
- Serialization functions (save_pickle_to_text and load_pickle_from_text) are included for saving and loading data.

main.py

- The main script orchestrating the project.
- It imports the DatasetGenerator and performs a search for each query, printing the results.
- Success rate statistics are printed at the end of the execution.

word_tokenizer.py

- The script contains the WordTokenizer class, which provides tokenization, punctuation removal, stopword removal, and lemmatization.
- It is used to tokenize sentences in the dataset.

Execution

To run the project, execute the main.py script. The search engine will process queries, calculate TF-IDF values, and print the search results.
Make sure to check the console for any additional output or error messages during execution.

Conclusion

This project demonstrates the implementation of a TF-IDF based search engine for document retrieval, showcasing the use of data structures and algorithms in information retrieval tasks.
Feel free to explore and modify the code for further experimentation and improvement.