Q1. Create a rudimentary Question Answering system that can answer questions related to the Israel Hamas war.

Project Overview

The goal of this project is to create a rudimentary Question Answering (QA) system that can answer questions related to the Israel-Hamas war using a dataset of news articles. Additionally, a timeline summarization plot or list can be created to show major events during a specific time period.

Step-by-Step Breakdown

Step 1: Understanding the Problem

Dataset Characteristics:

- The dataset contains 37,000 news articles.
- Articles span from October 2023 to March 2024.
- Not all articles are relevant to the Israel-Hamas war.
- The data might include noise like punctuation, special characters, and irrelevant symbols.

Project Requirements:

- Create a QA system for questions about the Israel-Hamas war.
- Optionally, create a timeline summarization of major events during the specified period.

Step 2: Data Preprocessing

Load the JSON File:

- Use Python's `json` library to load the dataset.
- Handle potential encoding issues by specifying the correct encoding (`utf-8`).

Explore and Clean the Data:

- Examine the structure of the dataset to understand the keys and format of the articles.
- Identify the correct keys for article content and titles (`articleBody` and `title`).
- Clean the article content to remove noise, such as punctuation and extra spaces.

Filter Relevant Articles:

- Filter articles that specifically mention both "Israel" and "Hamas" in their content.
- This ensures the QA system focuses on the relevant subset of articles.

Step 3: Model Setup and QA System

Choose a QA Model:

- Use a pre-trained QA model like `distilbert-base-uncased-distilled-squad` from Hugging Face's `transformers` library.
- This model is suitable for extracting answers from text based on input questions.

TF-IDF Vectorizer for Article Retrieval:

- Use `TfidfVectorizer` to convert articles and questions into numerical vectors.
- Compute cosine similarities to find the most relevant article for a given question.

Answer Generation:

- Implement a function that retrieves the most relevant article using TF-IDF and cosine similarity.
- Use the QA model to generate an answer from the retrieved article.

Step 4: Evaluation and Debugging

Debugging and Verification:

- Print intermediate outputs to verify that the filtering and processing steps are working correctly.
- Ensure that relevant articles are being correctly identified and processed.

Test the QA System:

- Test the QA system with example questions to evaluate the quality and relevance of the answers.

Key Decisions and Rationale

Using Pre-trained Models:

- Leveraging pre-trained models like BERT or DistilBERT allows us to build a functional QA system without training from scratch, saving time and computational resources.

Data Filtering and Cleaning:

- Filtering ensures that the QA system focuses on relevant content, improving the quality of the answers.
- Cleaning the data helps in reducing noise, which can improve the accuracy of the TF-IDF vectorizer and the QA model.

Using TF-IDF for Article Retrieval:

- TF-IDF (Term Frequency-Inverse Document Frequency) is a straightforward and effective method for text similarity, helping to identify the most relevant articles for a given question.

Debugging Outputs:

- Printing intermediate outputs aids in verifying each step, ensuring that the data is correctly processed and relevant articles are identified.

Potential Enhancements

Augmenting Data with External Sources:

- Incorporate additional information from sources like Wikipedia or DBpedia to enrich the dataset and improve answer quality.

Advanced NLP Techniques:

- Explore more advanced retrieval techniques like BM25 or embeddings-based retrieval for better relevance.
- Implement more sophisticated models like GPT-3 or custom-trained transformers for improved QA performance.

Timeline Summarization:

- Develop an additional module for extracting and summarizing key events into a timeline format, using NLP event extraction techniques.

Conclusion

This project involves preprocessing a large dataset, setting up a QA model, filtering relevant articles, and generating answers to questions about the Israel-Hamas war. Each step is designed to ensure the system is effective, efficient, and relevant, leveraging pre-trained models and simple text retrieval techniques to achieve the goal. Debugging and verification steps are crucial to ensure the process is working correctly, and potential enhancements can further improve the system's performance and usability.