Information Retrieval for Images

Group 99- Raghav Dunga, Salva Karimisahari, Benjamin Friedrich Richter, Branahvan Rajasooriar

Project Objectives & Scope

- Develop an AI-powered image retrieval system.
  - Implement a system where a user enters a query, and relevant images most similar to the query are retrieved
- Convert images into searchable information through automated labeling and vector embedding.
- Fine tuning to reach optimal prompts and embeddings
- Rank images based on query similarity.
  - Implement a ranking mechanism and sort images based on their “closeness” to a user query
- Evaluate retrieval effectiveness
  - Assess performance of system using established Information Retrieval metrics and manual labelling for validation

Why is efficient image retrieval important?

- The exponential growth of digital images demands efficient retrieval systems.
- Traditional image search methods often struggle with the semantic gap between image content and textual queries.
- Bridging the gap between low-level image features and high-level semantic descriptions.
- Ensuring that the retrieved images are both relevant and accurately ranked.

System Architecture Overview

High-Level Pipeline

- Data Acquisition & Preprocessing -
  - Acquire and preprocess images from a well-defined subset of COCO
- AI-Based Image Labeling -
  - Use a Gemini API to generate descriptive captions for images
- Embedding & Indexing -
  - Convert captions into fixed-size vector embeddings using Gemini Embedding
  - Index and store embeddings as well as other metadata in a vector database (Milvus).
- Query Processing & Retrieval -
  - Convert user queries into vector embeddings.
  - Perform nearest-neighbor search to retrieve and rank images by similarity or closeness to a query

Implementation Details

Project Structure -

- An overview of the repository layout -
    - Python modules for labeling (Gemini API integration), Hugging Face Integration, Caption Generation, Embedding Creation and Indexing, Query Processing
    - Web Application Modules - React.js, Node.js, TypeScript, CSS

Challenges and Solutions -

- Aligning AI-generated captions with the embedding model.
- Ensuring fast query response time through vector indexing.

Advantages

- Strengths -
  - Scalable Retrieval - Milvus enables efficient handling of large-scale embeddings.
  - Customizable Prompts - Allows flexibility in generating captions tailored to specific use cases.
  - Evaluation Metrics - Incorporates cosine similarity to assess caption quality.

Indexing Method & Sample Data

- Indexing Process -
  - Preprocessing -
    - Generate captions for images using the Gemini API.
- Creating the Index -
  - Convert text labels to high-dimensional vectors using an embedding model. Used the default embedding dimensionality for Google Gemini API - 3072
  - Store these vectors along with metadata (md5 hash keys, file path, captions) in Milvus.

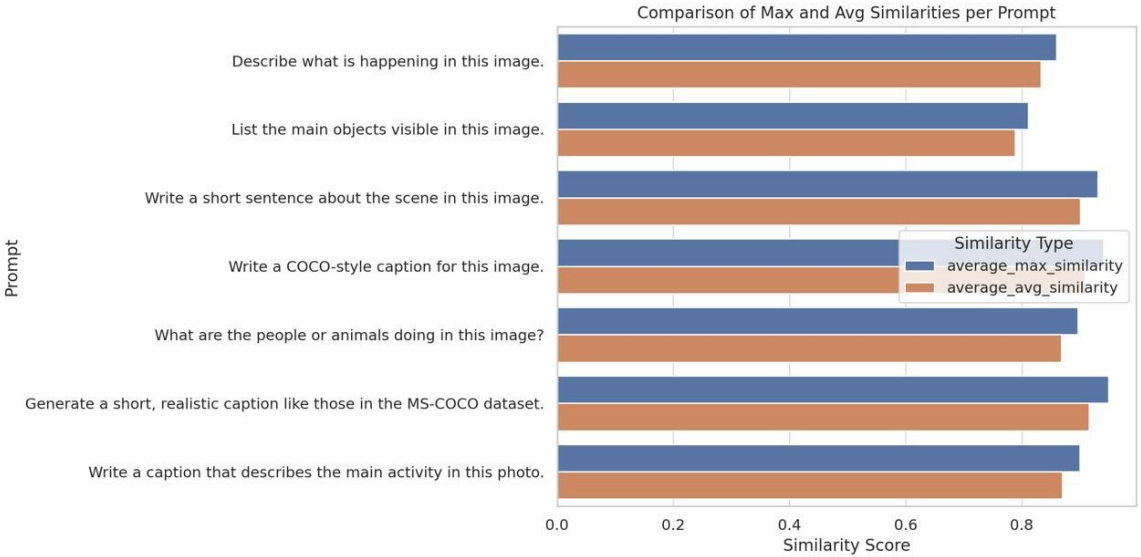
Information Retrieval Methods

- Data Preparation -
  - Created a subset of the data and saved locally to process efficiently. Created a utility file to handle this procedure.
- Labelling Images -
  - Used the Google Gemini API to label images and generate captions based on different prompts
  - Stored the results of the labelling - captions and corresponding metadata - md5 hashes, image paths, embeddings which are then stored in a SQLite Database for structured retrieval
- Text Embeddings -
  - Captions are converted into embeddings using Gemini's pre-built embedding model
  - Those embeddings are then stored in Milvus for retrieval later on based on similarity metrics

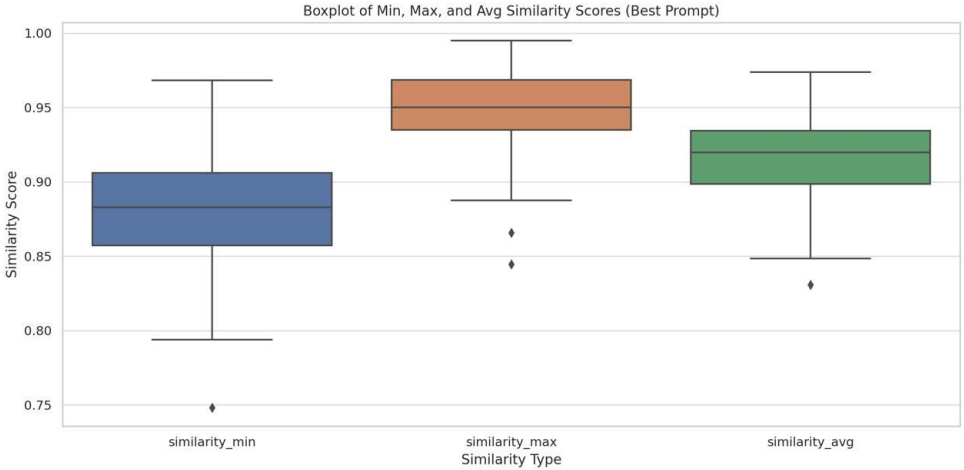
Ranking and Evaluation

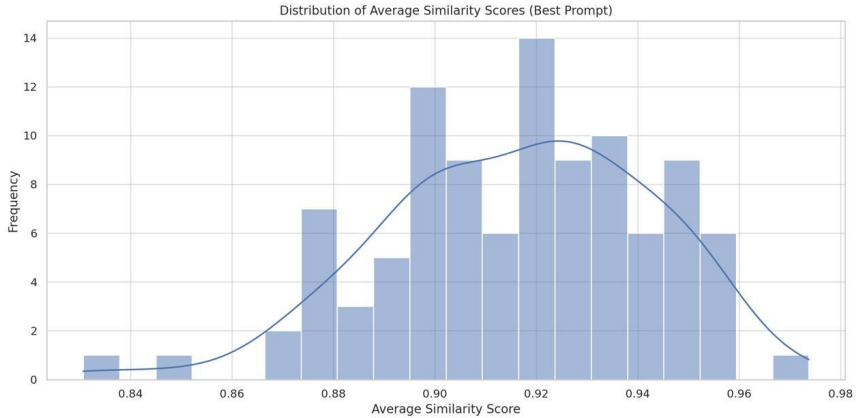
- Vector Similarity Search -
  - Similarity-based ranking done by Milvus using metrics like Cosine Similarity
  - Based on how close in proximity the embeddings are to an ideal result, results are ranked
- Cosine Similarity for Evaluation -
  - Captions generated by Gemini are compared to COCO reference captions using cosine similarity
  - Metrics implemented -
    - Maximum Similarity - Measures the closest match to reference
    - Average Similarity - Evaluates overall alignment with references
- Prompt-Level Evaluation -
  - Multiple template prompts were chosen and tested, results are then summarized for identifying the most effective prompt for caption generation.

Prompt-level similarity results

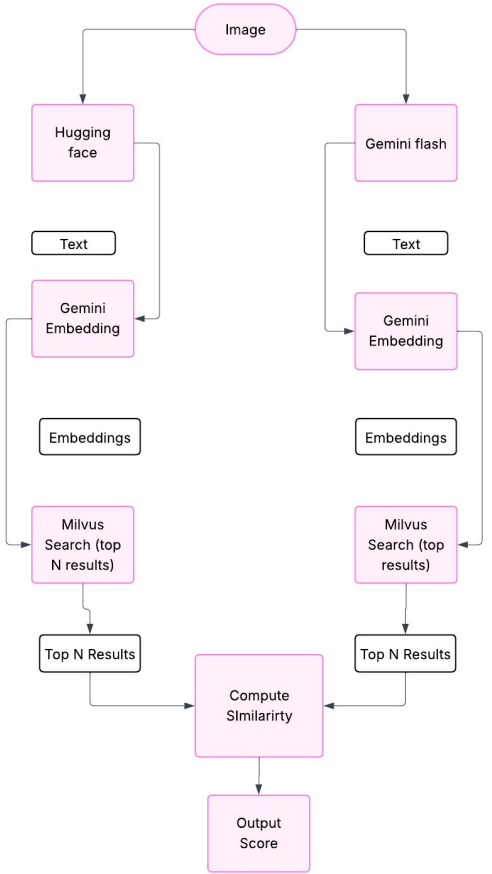


Zoom-in on the best prompt

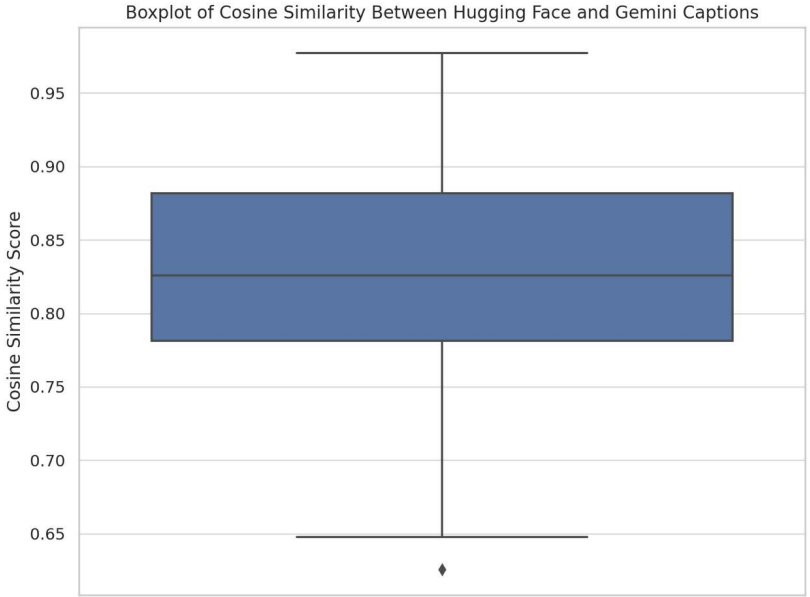


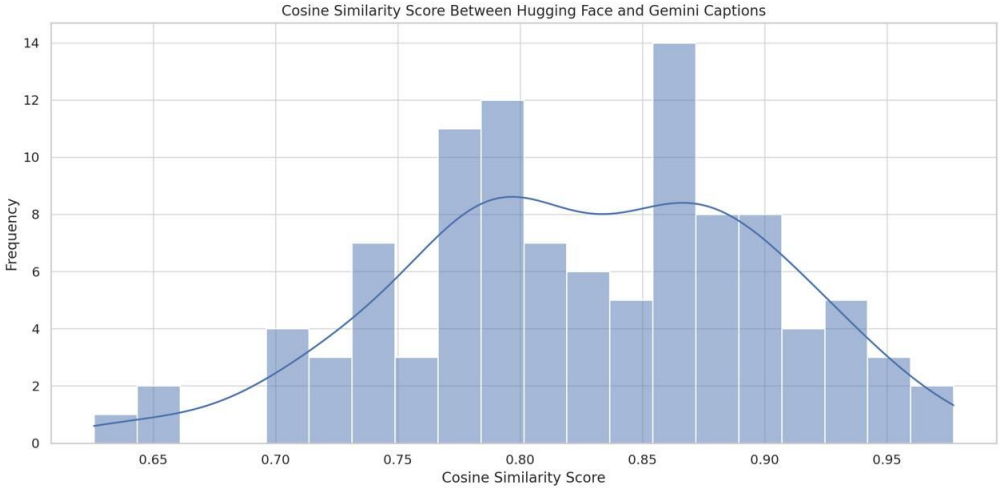


Comparing Gemini Embeddings to Hugging Face Embeddings

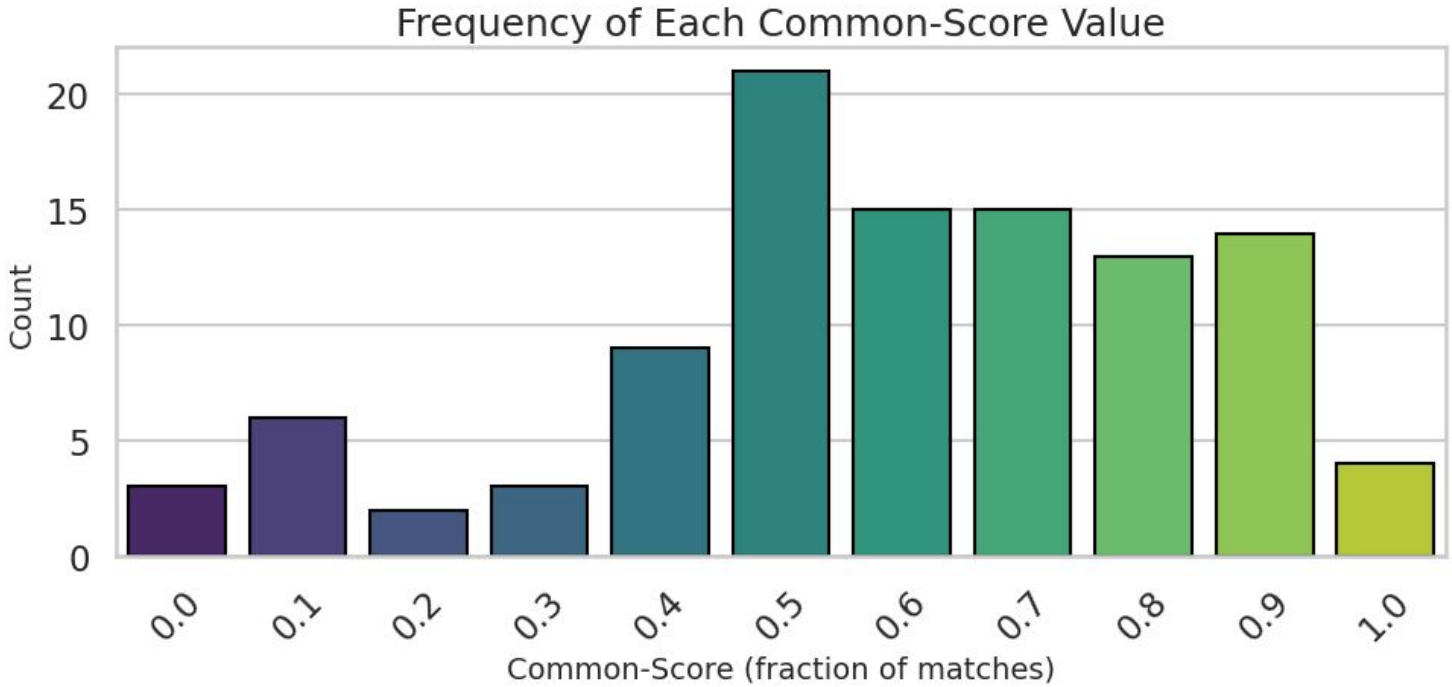


Embedding-level similarity results





Common-Score distribution



Conclusion & Future Work

Key Takeaways -

- Successfully built an image retrieval system with a focus on semantic search.
- Demonstrated the integration of AI labeling with vector-based indexing and retrieval.

Limitations -

- Num of queries
- API Resource Exhaustion

Future Enhancements -

- Integration of additional datasets to broaden the system's applicability.
- Experimentation with alternative embedding models for further accuracy improvements.
- UI enhancements and scalability improvements for real-world applications.