IR ASSIGNMENT 2 REPORT:

DATASET ANALYSIS :

Dataset consists of A2_Data.csv file 999 rows and 3 columns containing id, Image , Review Text.
Id : unique product id.
Image: list of urls for product.
Review Text : metadata/ review of product.

IMAGE DOWNLOADING :

Images downloaded using the urls and saved in Images folder. Some images failed to fetch.

IMAGE PREPROCESSING :

Basic image pre-processing techniques as altering contrast, resizing,geometrical orientation, random flips, brightness .etc applied on images and saved in preprocessed_input_image folder.

IMAGE FEATURES EXTRACTION:

For image feature extraction , VGG16 (pre-trained model for image feature extraction) used and saved image features are then dumped as pickle file .

TEXT PREPROCESSING :

The preprocessing pipeline involves below steps:

- Beautiful soup used for html content.
- Lowercasing:Convert the input text to lowercase using the lowercase_text function.
- Tokenization:Split the lowercase text into individual words or tokens using the tokenize_text function.
- Stopword Removal:Filter out common stopwords from the tokenized text using the remove_stopwords function.
- Punctuation Removal:Eliminate punctuation marks from the tokenized text using the remove_punctuations function.
- Blank Token Removal:Remove any remaining blank or empty tokens from the text using the remove_blank_tokens function.
- Stemming:Reduce each token to its root form using a stemming algorithm like the Porter Stemmer through the stem_tokens function.
- Lemmatization:Transform each token into its dictionary form using lemmatization via the lemmatize_tokens function.

- After preprocesses the Review text in each document (or image id) using the preprocess_text function and stores the preprocessed text in a new column called 'Preprocessed_Data'.

Text TF-IDF Calculation :

Preprocessing Documents: preprocess_text function used to preprocess the each review text corresponding to each image id/document and saved in Preprocessed Data].

Calculating Document Frequency (DF) and IDF (Inverse Document Frequency):Document frequency (document_frequency) is computed for each term in the unified vocabulary, representing the number of documents containing each term.

Inverse document frequency (idf) is calculated for each term using the formula idf = log(N / (document_frequency[word] + 1)), where N is the total number of documents.

Calculating TF-IDF Scores:For each document, the code iterates over the preprocessed text and computes the term frequency (term_frequency) for each term.

Then, it calculates the TF-IDF score for each term in the unified vocabulary using the formula tf_idf = (term_frequency[term] / len(terms)) * idf[term].

The TF-IDF scores are stored in a dictionary tf_idf_scores, with the document ID as the key and a dictionary of term-to-TF-IDF-score mappings as the value. Then it is saved as pickle file.

Image based Retrieval :

Input : image url and review text from user.

Input image downloaded in a folder , resized it and then passed it to a pre-trained VGG16 model . The extracted input image features are then saved in input_image_features[‘input_image’].

Loaded the saved features of the images in the corpus. Then cosine similarity scores calculated between input image feature and saved image features . The cosine similarities of all comparisons are saved in some data structure, then sort it and fetched top3 unique images . Corresponding review texts are fetched from A2_Data.csv. Then cosine similarity of the input review text and 3 review text fetched are calculated. The input review text tf-idf is calculated first and then compared with saved tf-idf of corpus. As result we got 3 (image,review text) pairs along with their cosine similarities. Now , for each pair composite similarity is calculated which is average of cosine similarities of image and text.

Output : image-review text pairs are ranked and displayed based on composite similarity.

Text based Retrieval :

Input : image url and review text from user.

Input image review text tf-idf calculated and compared it with saved tf-idf of corpus. Then all cosine similarities are sorted and saved in some data structure. Top3 review text fetched from sorted cosine similarities. The corresponding images of review texts are also retrieved from A2_Data.csv , then cosine similarity of input image and 3 fetched images are calculated. Composite similarity are calculated for each image-review text fetched above. Then ranking of pairs based on composite similarities.

Output : image-review text pairs are ranked and displayed based on composite similarity.

Result and Analysis :

The results in image based retrieval and Text based retrieval are save in some data structure for further analysis. Based on results , we concluded which technique is better and why.

As per my input query i am getting image based retrieval system as better than Text based retrieval system.

Choosing the best pre-trained model for image feature extraction can be challenging because some pre-trained models may take a significant amount of time to provide features, and they may generate large-sized feature representations, sometimes several gigabytes in size. Additionally, dealing with datasets containing URLs that fail to download images can introduce complications and affect the performance of the retrieval techniques.

Conclusion:

Despite facing challenges such as selecting the appropriate pre-trained model for image feature extraction and dealing with failed image downloads from URLs, the Image-based retrieval technique consistently outperforms the Text-based retrieval technique in terms of composite similarity scores. This superiority is attributed to the richness of representation in images, the ability of deep learning models to bridge the semantic gap effectively, and the contextual understanding provided by images. These factors collectively contribute to the Image-based retrieval technique's higher composite similarity scores, making it the preferred choice for similarity comparison in this scenario.