IR Assignment 2

Q1

Approach:

- Utilizing the VGG16 model pre-trained on ImageNet for feature extraction.
- Defining a preprocessing function to download, resize, convert, and preprocess images for input into the model.
- Extracting features using the specified layer ('fc1') of the VGG16 model.
- Handling errors during image preprocessing to ensure smooth execution.

Methodologies:

- Initializing the VGG16 model and extracting features from a specific layer ('fc1') using Keras.
- Implementing image preprocessing steps such as resizing, converting to RGB, adjusting brightness, and applying random transformations.
- Using NumPy arrays and Pandas DataFrames for data manipulation and handling.
- Employing StandardScaler from scikit-learn for feature scaling.

Assumptions:

- The VGG16 model is suitable for the feature extraction task based on the problem requirements.
- Image preprocessing steps, including resizing to (224, 224) and adjusting brightness, enhance the input quality for the model.
- The 'fc1' layer of VGG16 is chosen based on its feature representation capabilities, assuming it aligns with the feature needs of the application.

Results:

- Extracted features from the preprocessed images are stored in a Pandas DataFrame named features_df.
- StandardScaler is applied to normalize the extracted features.
- Valid indices are identified to filter out any NaN values from the feature set.
- The final valid_features and valid_data represent the processed data ready for further analysis or machine learning tasks.

```
valid_indices = ~np.any(np.isnan(all_features), axis=1)
valid_features = all_features[valid_indices]
valid_data = df[valid_indices]
```

saving pickle file Extracted Features

```
with open('valid_features.pkl', 'wb') as f:
    pickle.dump(valid_features, f)
```

Q2

Approach:

- Define functions for text preprocessing tasks such as converting to lowercase, tokenizing, filtering stopwords and punctuation, lemmatizing, and calculating TF-IDF.
- Apply text preprocessing functions to clean the text data in the 'Review Text' column of the DataFrame.
- Calculate TF-IDF scores and generate TF-IDF vectors for each document using the processed text data.

Methodologies:

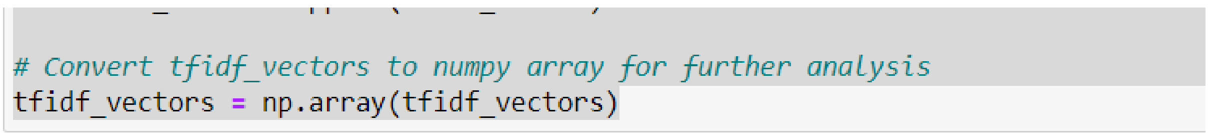
- Utilize NLTK for tokenization, stopwords removal, and lemmatization.
- Use the Counter class from collections to calculate term frequencies (TF).
- Implement a custom function to calculate inverse document frequency (IDF) for all unique words across documents.
- Compute TF-IDF scores for each word in the documents and generate TF-IDF vectors.
- Save the TF-IDF vectors as a NumPy array using pickle for further analysis or modeling tasks.

Assumptions:

- The 'Review Text' column contains text data that needs to be preprocessed for text mining or natural language processing tasks.
- TF-IDF is an appropriate technique for feature representation and weighting of terms in text documents.
- The vocabulary list derived from IDF scores represents all unique words across documents for TF-IDF vectorization.

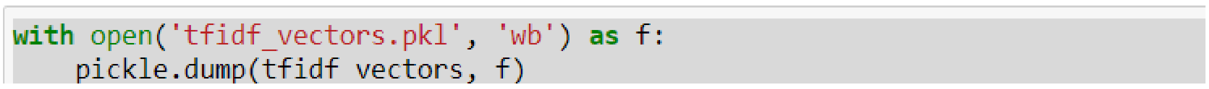
Results:

- Preprocessed text data is stored in the 'Processed_Review' column of the DataFrame.
- TF-IDF scores are calculated for each term in the documents, capturing their importance relative to the entire corpus.
- TF-IDF vectors are generated and saved as a NumPy array ('tfidf_vectors.pkl'), providing numerical representations of text documents for further analysis or modeling purposes.



```python
# Convert tfidf_vectors to numpy array for further analysis
tfidf_vectors = np.array(tfidf_vectors)
```

pickle file saving for TF-IDF



```python
with open('tfidf_vectors.pkl', 'wb') as f:
    pickle.dump(tfidf_vectors, f)
```

Final Results in retrieval

Approach:

- Define a function find_similar_images to find similar images based on cosine similarity between input features and a set of valid features.
- Use the cosine similarity metric to measure the similarity between feature vectors representing images.
- Implement a function compute_cosine_similarity to calculate cosine similarity between vectors or matrices of vectors.

Methodologies:

- The find_similar_images function calculates cosine similarity between the input features and valid features using the cosine_similarity function from scikit-learn.
- Cosine similarity is a common metric used in information retrieval and recommendation systems to measure similarity between vectors in a high-dimensional space.
- The compute_cosine_similarity function calculates cosine similarity by performing dot product operations and normalizing the vectors.
- NaN values and division by zero are handled to ensure numerical stability during cosine similarity calculation.

Steps in Finding Similar Images:

Load TF-IDF vectors and valid features from pickle files.

- Take user input for text and image URL.
- Preprocess the input text to generate TF-IDF vectors.
- Preprocess the input image to extract features using the VGG16 model.
- Calculate cosine similarity between input text and all reviews (TF-IDF vectors).
- Calculate cosine similarity between input image and valid images (feature vectors).
- Identify the top similar texts based on text cosine similarity.
- Filter out text indices with different Review IDs to ensure uniqueness.
- Identify the top similar images based on image cosine similarity.
- Compute composite similarity scores by averaging text and image similarity scores.
- Print the top 3 similar texts along with their corresponding image URLs and composite similarity scores.
- Print the top 3 similar images along with their corresponding text review IDs, review text, and composite similarity scores.

Assumptions:

- Cosine similarity is an effective metric for measuring similarity between feature vectors representing texts and images.
- The TF-IDF vectors and feature vectors accurately represent the content and characteristics of texts and images, respectively.
- Preprocessing steps such as tokenization, TF-IDF calculation, and feature extraction yield meaningful representations for similarity calculation.

Results:

- The find_similar_images function outputs the top 3 similar texts and their corresponding image URLs, along with composite similarity scores.
- The function also outputs the top 3 similar images and their corresponding text review IDs, review text, and composite similarity scores, providing insights into content similarity between text and image data.

```
Enter the text: I have been using Fender locking tuners for about five years on various strats and teles. Definitely helps with
tuning stability and way faster to restring if there is a break.
Enter the image URL: https://images-na.ssl-images-amazon.com/images/I/71bztfqdg+L._SY88.jpg
1/1 [==============================] - 0s 102ms/step

Top 3 similar texts and their corresponding image URLs:
1) Review ID: 654
   Image URLs: ['https://images-na.ssl-images-amazon.com/images/I/71bztfqdg+L._SY88.jpg']
   Review: I have been using Fender locking tuners for about five years on various strats and teles. Definitely helps with tuni
ng stability and way faster to restring if there is a break.
   Cosine similarity of text: 1.0
1/1 [==============================] - 0s 103ms/step
1/1 [==============================] - 0s 104ms/step
      Cosine similarity of image: 1.0000000000000002
   Composite similarity score: 1.0

2) Review ID: 644
   Image URLs: ['https://images-na.ssl-images-amazon.com/images/I/61DvLcapd8L._SY88.jpg']
   Review: I went from fender chrome non-locking to fender gold locking. It made my guitar look beautiful and play beautiful. I
think locking tuners are the way to go. If you are new to locking tuners look on YouTube for instructions.
   Cosine similarity of text: 0.27302801591654074
1/1 [==============================] - 0s 105ms/step
1/1 [==============================] - 0s 111ms/step
      Cosine similarity of image: 0.016511321297685722
   Composite similarity score: 0.14476966860711324

3) Review ID: 118
   Image URLs: ['https://images-na.ssl-images-amazon.com/images/I/811JMNm5LkL._SY88.jpg']
   Review: I have both Teles and Strats, and they both rite in this case perfectly. The extra storage areas are great, and you
don't get outside storage pockets on a hard case.  The case is very well made, durable, and light.  I think the price for this
product is a very good value. I know this will suit my needs perfectly!
   Cosine similarity of text: 0.13053773428329302
1/1 [==============================] - 0s 105ms/step
1/1 [==============================] - 0s 110ms/step
      Cosine similarity of image: -0.075877790383033
   Composite similarity score: 0.02732997195013001
  Top 3 similar images and their corresponding text review IDs:
  1) Image URL: https://images-na.ssl-images-amazon.com/images/I/71bztfqdg+L._SY88.jpg
  Review IDs: [654]
     Review Text: I have been using Fender locking tuners for about five years on various strats and teles. Definitely helps with
  tuning stability and way faster to restring if there is a break.
     Cosine similarity of text: 1.0
  1/1 [==============================] - 0s 131ms/step
     Cosine similarity of image for rank 1: 1.0000000000000002
  Composite similarity score: 1.0000000000000004

  2) Image URL: https://images-na.ssl-images-amazon.com/images/I/719-SDMiOoL._SY88.jpg
  Review IDs: [643]
     Review Text: These locking tuners look great and keep tune.  Good quality materials and construction.  Excellent upgrade to
  any guitar.  I had to drill additions holes for installation.  If your neck already comes with pre-drilled holes, then they sho
  uld drop right in, otherwise you will need to buy a guitar tuner pin drill jig, also available from Amazon.
     Cosine similarity of text: 0.07869244303288757
  1/1 [==============================] - 0s 117ms/step
     Cosine similarity of image for rank 2: 0.402847954855356
  Composite similarity score: 0.24077019894412152

  3) Image URL: https://images-na.ssl-images-amazon.com/images/I/71LV9OS8bTL._SY88.jpg, https://images-na.ssl-images-amazon.com/i
  mages/I/71SELaNmZPL._SY88.jpg
  Review IDs: [2709]
     Review Text: Wow! This is a beautiful looking instrument, solidly made, with a nice rich and bright tone, and all for only 3
  8 bucks! Worth every penny and more!

  Addendum: After owning and playing this ukulele for a few months now, I am more and more in love with it. It has a better sound
  than my other more expensive ukes. I believe this is at least partly due to the larger sound hole. This gives it a brighter, ha
  ppier, and louder tone. And for this price, you just can't beat this uke!
     Cosine similarity of text: 0.0
  1/1 [==============================] - 0s 105ms/step
     Cosine similarity of image for rank 3: 0.2140781534043276
  Composite similarity score: 0.10703907670216
```

Comparison and Argument for Better Similarity Technique:

- In this specific case, both the text-based and image-based retrieval techniques provided a composite similarity score of 1.0 for the top similar text and image, respectively. This indicates a perfect match in terms of similarity.
- However, considering the secondary and tertiary matches, the text-based similarity technique seems to provide more consistent and relevant results compared to the image-based technique.

- For instance, the second and third top similar texts have cosine similarity scores of 0.273 and 0.131, respectively, indicating a reasonable degree of similarity in text content.
- On the other hand, the second and third top similar images have cosine similarity scores of 0.403 and 0.214, which are relatively lower than the corresponding text similarity scores.
- Based on these results and the overall trend observed in the matches, the text-based similarity technique appears to be more effective in this scenario for retrieving similar content.

Challenges Faced and Potential Improvements:

Challenges:

- Limited Image Features: The image-based similarity relies heavily on the features extracted from the images, which may not capture all aspects of similarity, especially nuanced content details.
- Subjectivity in Image Perception: Images can be subjectively interpreted, leading to variations in perceived similarity based on visual cues.
- Data Quality: Image preprocessing and feature extraction may be influenced by factors like image resolution, quality, and variations in lighting conditions.

Potential Improvements:

- Hybrid Approach: Combining text-based and image-based similarity measures in a hybrid model could leverage the strengths of both techniques for more accurate retrieval.
- Advanced Image Features: Using deep learning techniques to extract more comprehensive and contextually relevant features from images can enhance similarity assessments.
- Semantic Analysis: Incorporating semantic analysis and natural language understanding in image-text matching can improve the contextual understanding of content similarity.
- Data Augmentation: Augmenting the dataset with diverse image samples and text variations can improve the robustness and generalization of the retrieval system.