Similarity of PDF having both Text and Images.

Similarity of PDF with text.(text_similarity.py)

- First, we have taken pdfs containing ‘Terms & Uses’ from some random websites, stored them in a folder and ran a python script to load them, then started reading each file using PyPDF2 module and stored them in a dataframe in lower case which then stored as a pickle file.
- Next, we have implemented a series of different techniques to get an idea of how the values of similarity indexes for any 2 files using various methods like Levenshtein distance, Cosine similarity, Jaccard similarity, and Hamming distance are looking like with the help of libraries like Levenshtein, textdistance, sklearn.
- With the libraries like PorterStemmer, NLTK; the preprocessing on the previous dataframe which contained lowercase letters has been done with removing non-alphanumeric characters, tokenization, eliminating stopwords, and stemming, which then stored as a pickle file.
- Now after including the above preprocessing steps, the whole code has rerun once again. Now, a csv file has been generated with columns having the names of the 2 selected pdfs, similarity value of a method without preprocessing happened and similarity value of a method with preprocessing happened

Similarity of PDF with images between text.(image_text_similarity.py)

- Here with the help of PyPDF2 library we loaded the pdfs and extracted the text then made it into lowercase and saved it in a dataframe and with the help of fitz library, extracted the image from a page of the pdf which then stored in a folder (ex. images).
- So after the images have been extracted, we have done preprocessing of the image using grayscale, gaussian blur, histogram equalization resizing and normalizing pixel values with the help of cv2.
- Then we calculated the image similarity between 2 images using SSIM and MSE from skimage.matrices library.
- Simultaneously, we calculated the similarity value between the text using the same techniques like the above and we are saving it into a csv file.
- After going through the whole process for the image, we are saving the images and the similarity value into the dataframe and finally saving it into a different csv file naming ends with ‘_image’

Streamlit integration to display similar images of the selected image.

- Here, we directly called a function from the image similarity checking function from the image_text_similarity.py so as to obtain the images to display in the streamlit.
- We can select the name of a photo from the set of images from extract_images such that the selected image will be displayed on the left side and similar images along with their similarity values are displayed on the right side using st.columns