File Similarity Detection

Introduction:

In this take, we focus on the extraction, pre-processing, clustering, and classification of text data from various file formats such as PDF, DOCX, and TXT. By leveraging Natural Language Processing (NLP) techniques and machine learning algorithms, the goal is to categorize documents based on their content, ensuring seamless organization and retrieval. The process involves converting the textual data into structured formats, cleaning and normalizing the text, and then using word embeddings to represent the text in a numerical form. Clustering algorithms are applied to group similar documents, while classification models are used to label the documents accurately. This comprehensive approach aims to enhance document management systems, making them more efficient and intelligent.

Step-by-step Approach:

Install Required Libraries:

- The code installs necessary libraries: PyPDF2, python-docx, contractions, and unidecode.

Import Libraries:

- The code imports various libraries for handling PDFs, DOCX files, text processing, and machine learning.

Define Functions for Reading Files:

- Functions store_pdf_data and store_doc_data are defined to read content from PDF and DOCX files respectively.

Extract Content from Files:

- The code iterates through files in a specified directory (/content/Data), reads content from PDF, DOCX, and TXT files, and stores the content along with metadata (filename, file type, label) in a list. This list is then converted to a pandas Data Frame.

Text Cleaning:

- Functions for downloading necessary NLTK resources and for cleaning text are defined and applied. The text cleaning steps include:
  - Expanding contractions.
  - Converting text to lowercase.
  - Removing non-alphabetical characters.
  - Decoding text.
  - Tokenizing text.
  - Removing stopwords.
  - Lemmatizing tokens.

    - Label Encoding: Labels are encoded into numerical values for classification purposes.

Train-Test Split:

- The cleaned data is split into training and testing sets.

Word2Vec Embedding:

- A Word2Vec model is trained on the training data to create word embeddings.
- The trained model is saved for future use.
- Sentence vectors are created by averaging the word vectors for words in each sentence.

Determine Optimal Number of Clusters:

- The Elbow Method and Silhouette Method are used to determine the optimal number of clusters for KMeans clustering.

KMeans Clustering:

- KMeans clustering is performed with the determined optimal number of clusters (3 in this case).
- The data points are assigned to clusters, and the distribution of data points across clusters is printed.
- The Silhouette Score is calculated to evaluate the clustering quality.

Classification:

- Three classifiers are defined: Logistic Regression, Decision Tree, and Random Forest.
- Each classifier is trained and evaluated on the training and testing sets.
- The evaluation metrics (Accuracy, Precision, Recall, F1 Score) for each classifier are calculated and compared.
- Results are plotted to visualize the performance of each classifier.
- It is concluded that the Random Forest classifier performs the best among the three.

Challenges:

- *Dataset acquisition:*

  The major challenge encountered in this task is the creation of a cohesive dataset, as the data has been sourced from various websites. This collection process ensures that only certain files have matching contextual information.

- *Various file formats handling:*

  Handling different file formats (PDF, DOCX, TXT) requires specialized methods for data extraction, as each format has its unique structure and encoding.

- *Tuning the hyperparameters:*

Ensuring high accuracy, precision, recall, and F1 scores for classification models involves selecting the right algorithms and tuning hyperparameters, which can be time-consuming.

Future improvements:

- *Advanced text extraction techniques:*

Invest in research and development of advanced text extraction techniques specifically tailored for handling complex layouts, images, and tables within PDFs. This could involve leveraging computer vision algorithms for better text extraction accuracy.

- *Advanced word embeddings:*

BERT word embeddings can be considered as this captures the contextual information based on the whole document. These embeddings often outperform traditional word embeddings like Word2Vec in capturing semantics and context.

- *New hyperparameters consideration:*

Some of the hyperparameters can be explored in order to improve the model’s performance.