REPORT

Introduction

This report describes the efforts made to develop a library that can recognize and split various documents such as PAN, Aadhaar, Bank Statement, ITR/Form 16, Customer Photograph, Utility Bill, Cheque Leaf, Salary Slip/Certificate, Driving License, Voter ID, and Passport, that are present in a single file (image/pdf/word document) submitted by the user. The library also extracts the data from the split documents.

Document Splitting

The first step in the development of the library was to create a document splitter that could identify and classify the documents present in a single file. To do this, Optical Character Recognition (OCR) techniques such as Tesseract and Poppler were used to extract text from the input file. The extracted text was then pre-processed and used as input to train a machine-learning model.

A convolutional neural network (CNN) was chosen as the machine learning model for document classification due to its ability to learn features from image data. The CNN was trained on a dataset of various documents that were manually labelled according to the document type. This dataset was made my personal informations so couldn't be shared. This can be seen in the file document_splitter.py.

Data Extraction

Once the documents were split and classified, the next step was to extract relevant information from the split documents. For this, techniques such OCR was used and then the extracted text was saved. This can be seen in the file extractor.py.

Evaluation

The trained model was tested on a separate dataset, however, as the dataset used for training was not very large, the model's accuracy was not very high. There was a case of bias and overfitting due to lack of many instances in my dataset. To improve the model's accuracy, it is suggested to train it with a larger and more diverse dataset. I used dropouts to reduce overfitting, if trained properly on large dataset then it will show high accuracy.

Implementation and Deployment

The code is developed in python and the user can run the code by providing the input file, path of tesseract and poppler and the code will output a csv file with the column names name, text and label where the name is the file name of the test file uploaded, the text is the text in that file, the label is the class to which it belongs to(PAN, Aadhar.....). All the architecture has been done in train.py just make sure you upload the data in data/train in the folders of the class names.

Steps to run the code:

1) To run this file make sure you upload a pdf containing multiple pages or an image in jpeg format or a word document in data/test/predict
2) Alternate (Train the model using a good set of dataset and store the weights. All the architecture has been done in train.py just make sure you upload the data in data/train in the folders of the class names.)
3) Make sure you configure the path of tesseract and poppler in extractor.py and document_splitter.py respectively.
4) You just have to run the main.py by the command
python main.py
5) It will output a csv with the column names name,text,label where name is the file name of the test file uploaded , text is the text in that file, label is the class to which it belongs to (PAN,Aadhar.....).

Further Improvements

1) In future, there are a few areas that can be improved for better performance and to increase the accuracy of the model
2) Use of more advanced OCR technologies such as Google Vision API or Microsoft Azure OCR
3) Incorporation of Natural Language Processing (NLP) techniques to improve the data extraction process.
4) Use of a more diverse dataset for training the model

Conclusion

In conclusion, the developed library utilizes Optical Character Recognition (OCR) techniques to extract text from input files, which is then used to classify the documents into various categories such as PAN, Aadhaar, Bank Statement, ITR/Form 16, Customer Photograph, Utility Bill, Cheque Leaf, Salary Slip/Certificate, Driving License, Voter ID, and Passport. The document splitter component of the library separates the multiple documents present in a single file into individual pages, and the data extractor component extracts relevant information from the split documents. However, to improve the model's accuracy, it is suggested to train it with a larger and more diverse dataset.