**Report**

## Aim: Classify given pdf urls into 4 categories - fuses, cable, lighting, others.

## Code availability: [Drive link](https://drive.google.com/drive/folders/1T5ZlUWQFmyeaRxMTty09bU4RhLZ9aveO?usp=drive_link)

## Introduction:

Various OCR libraries are used to extract text from PDF. I choose to extract from url itself instead of downloading to reduce space complexity. On the other hand, if the [PDF](https://html5.dcatalog.com/?docid=8e9daddd-82b0-4ed4-a656-a8aa011ea6d3#page=153) is embedded in a flipbook or PDF-viewer, the second page of the document is downloaded and processed for text extraction because the first page includes the instrument's brand name and minimum details.

## Requirements:

**OCR and PDF**: PyPDF2, pytesseract, Pillow, PyMuPDF

**Embedding:** Sentence-transformers

**Data Manipulation**: pandas, numpy

**Visualization**: matplotlib.pyplot

**Web Scraping**: BeautifulSoup

**Machine Learning**: scikit-learn, xgboost

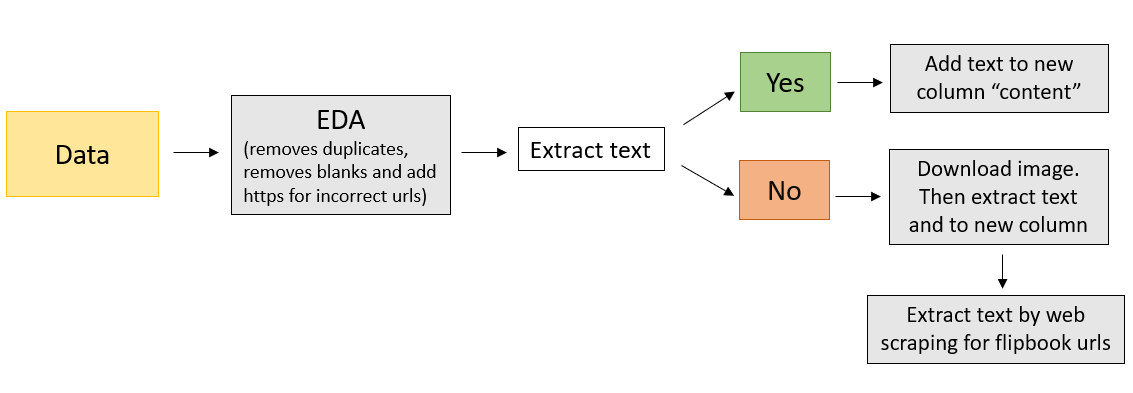
**Utilities**: os, re, io, time, concurrent.futures

## Data extraction process:

The dataset initially consisted of **1,845** entries. However, after conducting Exploratory Data Analysis (EDA) and applying various data cleaning techniques, the dataset was reduced to **1,262** rows. Some pdf urls were embedded in pdf-viewer. Upon closer inspection of the webpage, the og:image meta tag was found to contain images of each page of the PDF. From here, the image was downloaded, and text was extracted.

Some urls in train data were invalid and showed 404 error.

After this process, training was performed on a total of **1035** datapoints.



## Models:

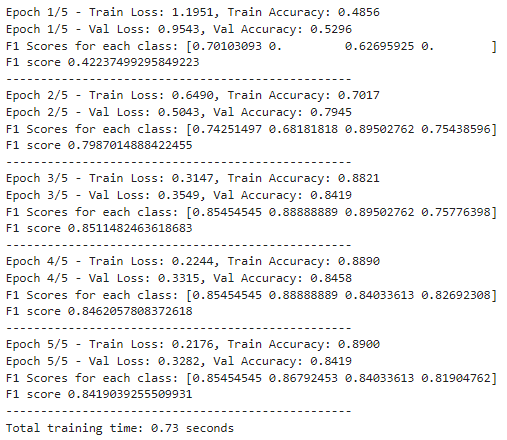
Two models were trained on this prepared dataset - a Deep learning model and a Machine Learning model. Both models performed similarly during testing; however, the training accuracy was found to be lower in the ML model.

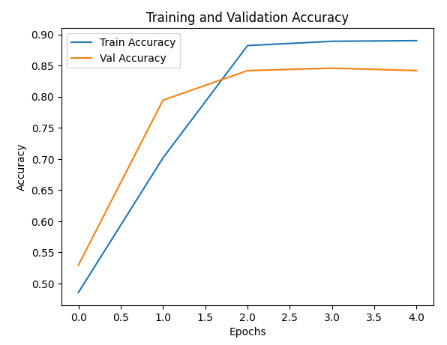
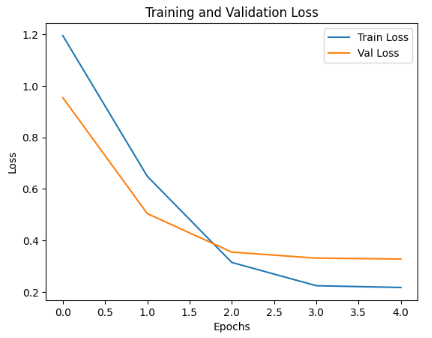
1. ***DL model:***

A simple ANN architecture was used with one hidden layer of 128 neurons for the DL model. To avoid overfitting, a dropout layer was added. However, after this layer, testing performance was reduced. Thus, to overcome this problem, the number of epochs was reduced.

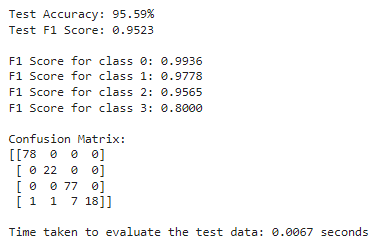
Logsoftmax was used as the final layer, and the Negative Log Likelihood loss function was used while training.

For evaluation, the f1 score for each class, accuracy, and confusion matrix were printed.

****

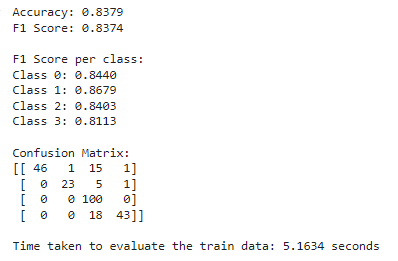


**Testing the DL model**

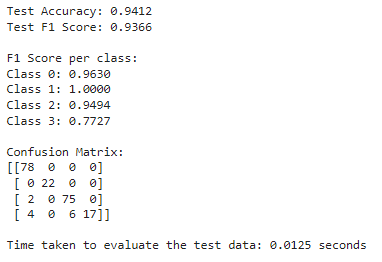


1. ***ML model:***

XGBoost was used for the multi-class classification task. The same evaluation metric was used.



**Testing the ML model**

****

## Conclusion & Future Prospects:

1. Both the ML and DL models performed almost the same during evaluation (testing data).
2. There could be some scope for improvement while processing the pdf. Example: Some links have downloadable pdf URLs that are not processed by the above pipeline.