Deep Learning – Lab Project Report

Udit Amin - 200968132

Problem Statement :

Given a set of grayscale document images, the task is to classify each image into one of the 16 classes or document types. The training dataset consists of 16000 images with 1000 images belonging to each class. Example images are provided below for some classes. The dataset was collected from the RVL-CDIP dataset [1].

Objectives :

- Replicate the original DocFormer and LayoutLM models
- Achieve an accuracy threshold of atleast 75%
- Real life implementation and deployment into a cloud service using Docker

Metrics :

- Accuracy
- Precision
- Recall
- F1-score
- Confusion Matrix
- ROC AUC

Meta-data :

The RVL-CDIP dataset consists of scanned document images belonging to 16 classes such as letter, form, email, resume, memo, etc. The dataset has 320,000 training, 40,000 validation and 40,000 test images. The images are characterized by low quality, noise, and low resolution, typically 100 dpi.

The data used for this problem statement is based on the above mentioned dataset.

The dataset at hand has document images, categorized into train and test directories. These directories contain image (.tiff) files. There is also a file for labels which has id and it's corresponding label for mapping.

The dataset contain 16 classes of documents, which are :

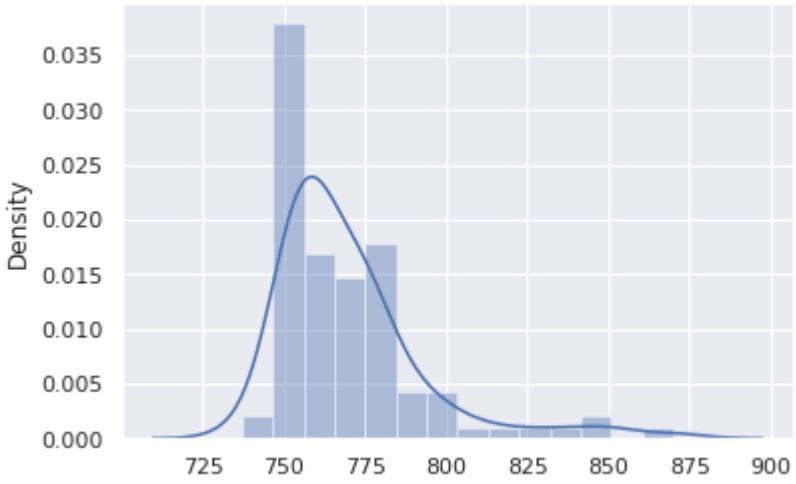
1 : memo
2 : form
3 : email
4 : handwritten
5 : advertisement
6 : scientific report
7 : scientific publication
8 : specification
9 : file folder
10 : news article
11 : budget
12 : invoice
13 : presentation
14 : questionaire

15 : resume
16 : letter

Reference Paper

Exploratory Analysis :

The dataset is clean as it's been provided on kaggle. There isn't much analysis apart from the images themselves.



The above distplot provides the class distributions for the different classes.
There is a class imbalance which can have an impact on the performance of the classifier.

While outputting the different images from each class shows that some classes like files and folder doesn't have that well defined images, the reason could be the way the images were processed.

Pre-processing Pipeline :

For the baseline model, we are using CNN models. So for that we will be using the required tasks like applying convolutinal kernels for extracting feature map, and experiment with it and the pooling operations. Further we will be moving along with experimentations with different models like transformer networks and OCRs.

Approach:

The topic of document page image classification has received much publicity over the last few years. In fact the RVL-CDIP dataset was curated specifically to test image classification strategies on document images. Earlier studies focused heavily on the original AlexNet2 architecture1,3. More recently modern architectures such as VGG16, GoogLeNet5, and ResNet50 have been proposed and tested on RVL-CDIP. The current state-of the-art utilizes a set of 5 distinct VGG16 models, one for the whole image (known as the holistic model, initialized with pretrained ImageNet weights) and 4 for specific subsections of the image (header, footer, left body, and right body initialized on

the holistic trained weights). These 5 models are then combined to form a final prediction. While accurate, the number of parameters is immense (on the order of 10^8) and the training process is sequential, requiring a holistic model to be trained before any of the subsection models can be trained.

In addition to the aforementioned image classification strategies, we can take advantage of optical character recognition (OCR) technology to extract text from document page images and train text classification algorithms. Many approaches have been developed to deal with text classification problems, although most have been developed under the assumption of clean encoded text. There is evidence to support that the bag-of words approach is quite robust to the unavoidable transcription errors.

In this project we explore combining both approaches into a single classification task, i.e. we construct a model that uses both the visual information and the textual content of page to make a decision.

We also explore the possibilities of including LayoutLM model for pos embedding and drawing bounded boxes to extract special features using NLP, as well as a multi-modal Architecture based on VGG16 and exploring the results.

Reference publications:

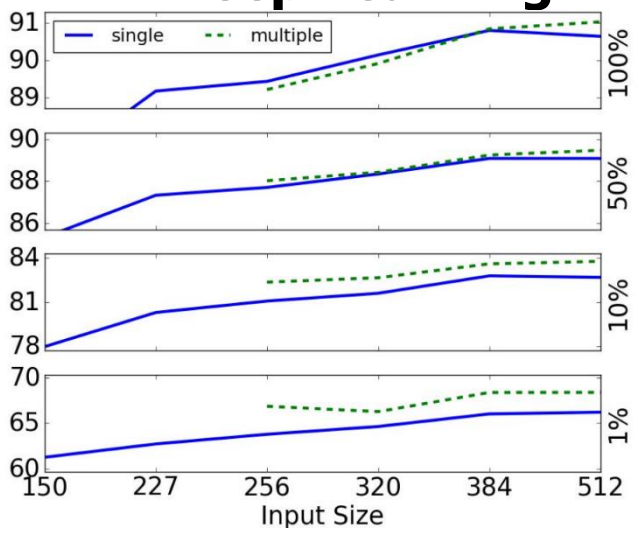
a. Analysis of Convolutional Neural Networks for Document Image Classification
b. Modular Multimodal Architecture for Document Classification.
c. Document Image Classification with Intra-Domain Transfer Learning and Stacked Generalization of Deep Convolutional Neural Networks
d. DocFormer: End-toEnd Transformer for Document Understanding

Baseline Transfer-Learning CNN models:

Hyperparameters:
We empirically determined good training schemes for each dataset. For RVL-CDIP, CNNs are trained with Stochastic Gradient Descent (SGD) with mini-batches of 32 for 500,000 weight updates. The initial learning rate (LR) is 0.001 and is decayed by a factor of 10 every 150,000 minibatches. We also use Momentum optimizer with Beta set to 0.9.
All networks are trained on the training split and progress is monitored on the validation set. The set of network parameters that performed best on the validation set is then evaluated on the test images. In this work, we use test set accuracy as our evaluation metric. CNNs require days to train on high-end GPUs, so training many CNNs for statistical significance testing is often too time-consuming. For this reason, CNN literature almost always reports numbers from only a best single trained model. Thus we typically report average performance of 1-2 CNNs.

Input Shape:
Standard CNN architectures accept inputs that are 224x224, but to our knowledge, this design. choice has not been thoroughly explored before. At this resolution, large text is generally legible, but smaller text is not. We empirically test square input sizes {n × n|n = 32, 64, 100, 150, 227, 256, 320, 384, 512}. Of necessity, we modify the architecture of the CNN for each input size with the following principles.
1) Same number of layers
2) Increase kernel size and network width with input size
3) Spatial output size of final convolution layer is 6x6

Deep Learning – Lab Project Report



Accuracy vs input size. Solid blue lines are for CNNs trained with a single input size. Dashed green lines are for CNNs trained with variable sized input images. The x-axis location corresponds to the largest input size used to train the CNN. For example, the performance of the CNNs trained with images of size 256-384 is plotted at x=384.

An exhaustive grid search over possible architectures was not possible with our computational resources

All models have been evaluated based on the accuracy of the output and Loss.

Tested Architectures:

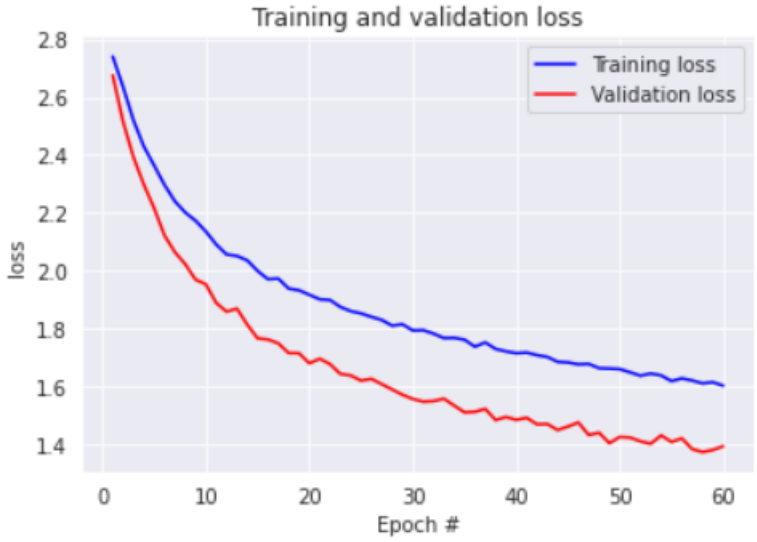
a) **GoogleNet:**

Over 20 epochs, Training accuracy: 47.57%, val accuracy: 51.07%

Deep Learning – Lab Project Report

b) **ResNet50:** Over 60 epochs, Training accuracy: 61.23%, val accuracy: 58.25%

c) **VGG16:**
Over 60 epochs, Training accuracy: 49.59%, val accuracy: 61.20%



Result:
While CNN architectures by themselves give a decent performance, it is nothing to write home about. Incorporating the extracted text from the documents using OCR however, should give us a better result. This may give the models an insight into the actual content of the documents and in theory should yield better accuracy and lower loss.

Deep Learning – Lab Project Report

DocFormer: Multi Modal Approach:

Model Architecture DocFormer is an encoder-only transformer architecture. It also has a CNN backbone for visual feature extraction. All components are trained end-to-end. DocFormer enforces deep multi-modal interaction in transformer layers using novel multi-modal self-attention. We describe how three modality features (visual, language and spatial) are prepared before feeding them into transformer layers.

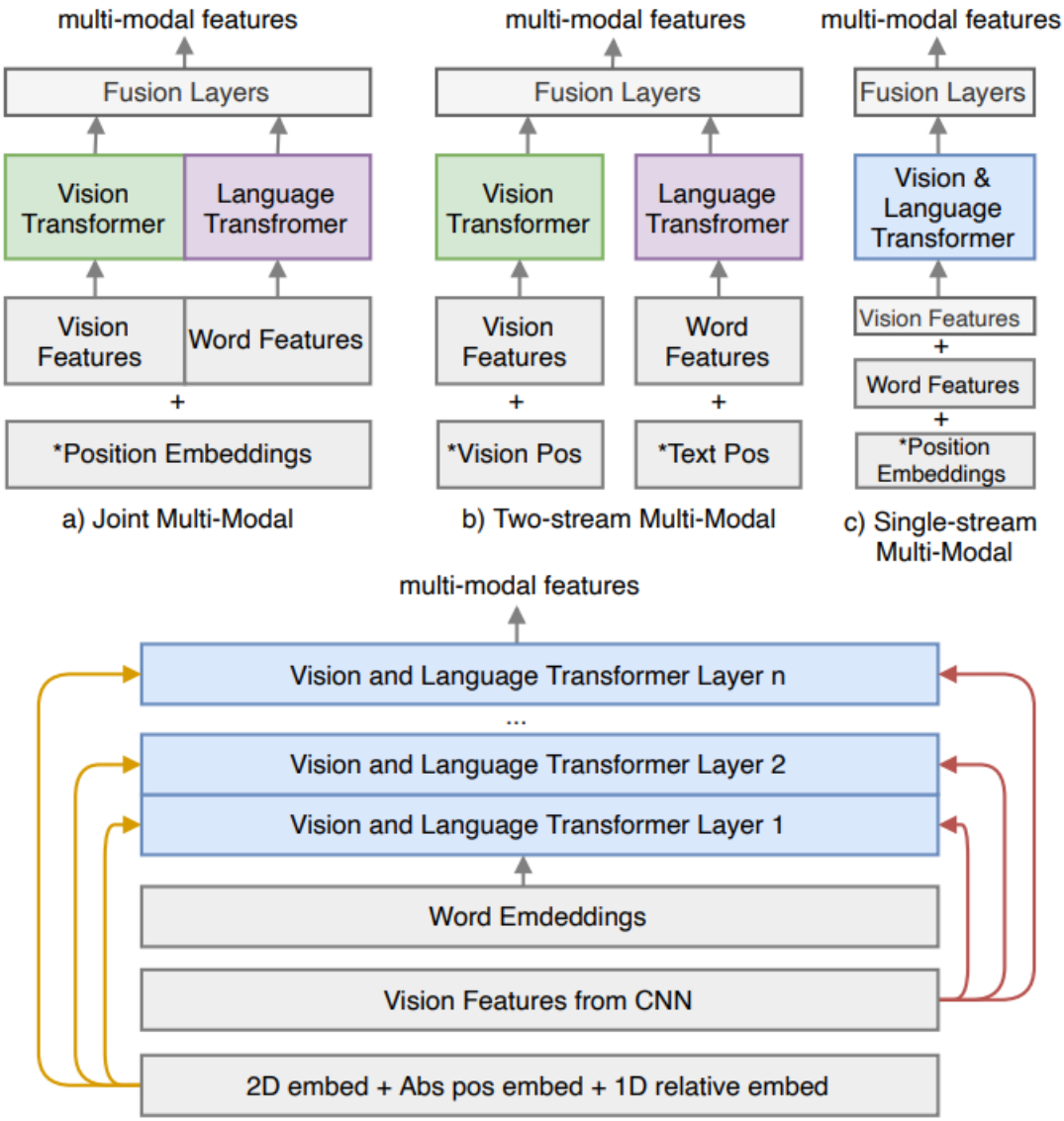
Visual features: Let v ∈ R 3×h×w be the image of a document, which we feed through a ResNet50 convolutional neural network fcnn(θ, v). We extract lower-resolution visual embedding at layer 4. Typical values at this stage are c = 2048 and hl = h 32 , wl = w 32 (c = number of channels and hl and wl are the height and width of the features). The transformer encoder expects a flattened sequence as input of d dimension. So we first apply a 1 × 1 convolution to reduce the channels c to d. We then flatten the ResNet features to (d, hl × wl) and use a linear transformation layer to further convert it to (d, N) where d = 768, N = 512. Therefore, we represent the visual embedding as V = linear(conv1×1(fcnn(θ, v))).

Language features: Let t be the text extracted via OCR from a document image. In order to generate language embeddings, we first tokenize text t using a word-piece tokenizer to get ttok, this is then fed through a trainable embedding layer Wt. ttok looks like [CLS], ttok1 , ttok2 , . . . , ttokn where n = 511. If the number of tokens in a page is > 511, we ignore the rest. For a document with fewer than 511 tokens, we pad the sequence with a special [PAD] token and we ignore the [PAD] tokens during self-attention computation. We ensure that the text embedding, T = Wt(ttok), is of the same shape as the visual embedding V . Following, we initialize Wt with LayoutLMv1 pre-trained weights.

Spatial Features: For each word k in the text, we also get bounding box coordinates bk = (x1, y1, x2, y2, x3, y3, x4, y4). 2D spatial coordinates bk provide additional context to the model about the location of a word in relation to the entire document. This helps the model make better sense of the content. For each word, we encode the top-left and bottom-right coordinates using separate layers Wx and Wy for x and y-coordinates respectively. We also encode more spatial features: bounding box height h, width w, the Euclidean distance from each corner of a bounding box to the corresponding corner in the bounding box to its right and the distance between centroids of the bounding boxes,

Multi-Modal Self-Attention Layer: We now describe in detail our novel multi-modal self-attention layer. Consider a transformer encoder fenc(η, V , Vs, T , Ts), where η are trainable parameters of the transformer, V , Vs, T and Ts are visual, visual-spatial, language and language-spatial features respectively, and are obtained as described previously. Transformer fenc outputs a multi-modal feature representation M of the same shape d = 768, N = 512 as each of the input features. Self-attention, i.e., scaled dot-product attention, for a single head is defined as querying a dictionary with key-value pairs.

Deep Learning – Lab Project Report



Document Classification Task with DocFormer

For this task we use pooled features to predict a classification label for a document. The RVL-CDIP dataset consists of 400,000 grayscale images in 16 classes, with 25,000 images per class. However due to resource limitations, we were only able to train the model with a smaller subset of the dataset. Overall there are 9,000 training images, 3,000 validation images. We report performance on test and eval metric is the overall classification accuracy. In line with prior text and layout information is extracted using Textract OCR.

```
config = {
  "coordinate_size": 96,              ## (768/8), 8 for each of the 8 coordinates o
f x, y
  "hidden_dropout_prob": 0.6,
  "hidden_size": 768,
  "image_feature_pool_shape": [7, 7, 256],
  "intermediate_ff_size_factor": 4,
```

Deep Learning – Lab Project Report

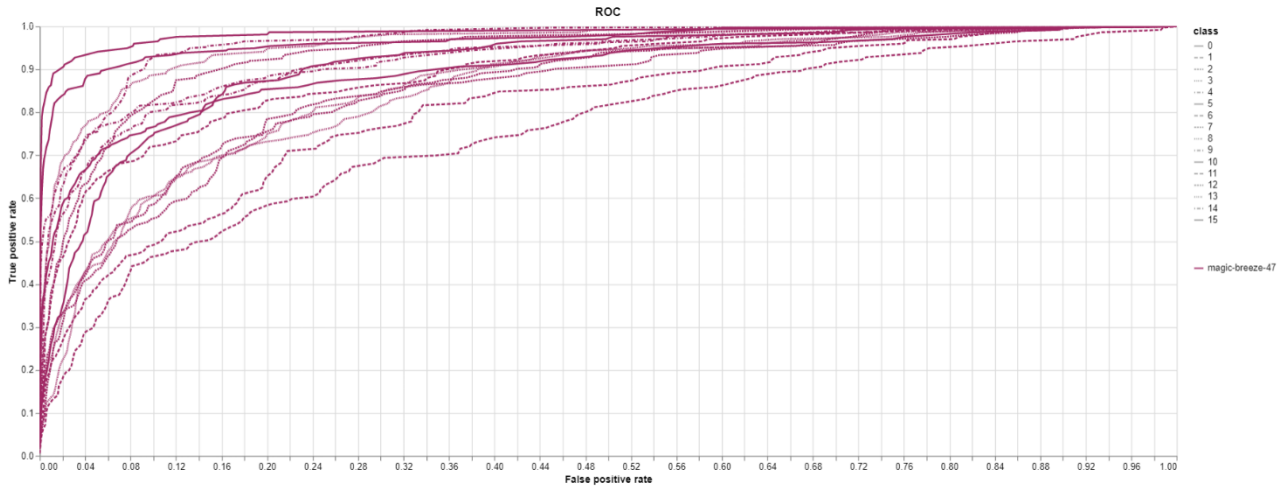

```
  "max_2d_position_embeddings": 1024,
  "max_position_embeddings": 128,
  "max_relative_positions": 8,
  "num_attention_heads": 12,
  "num_hidden_layers": 3,
  "pad_token_id": 0,
  "shape_size": 96,
  "vocab_size": 30522,
  "layer_norm_eps": 1e-12,
}
```


Observations:

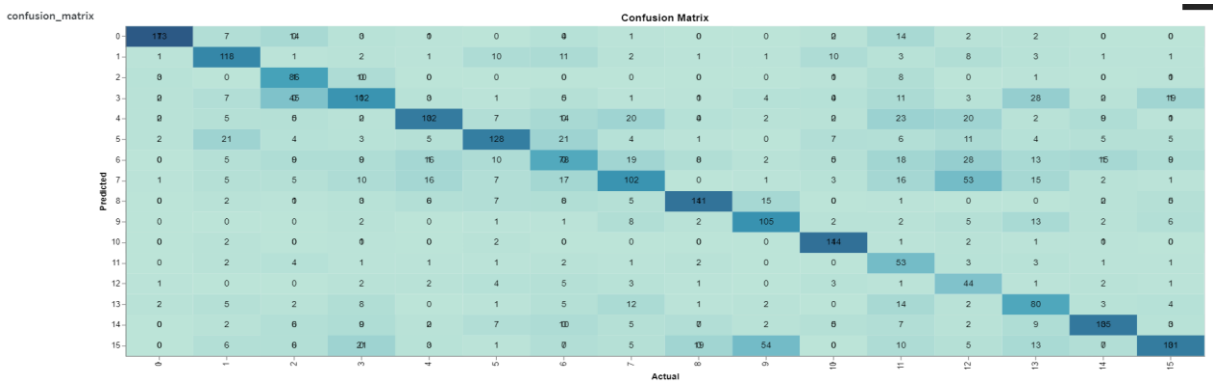
Model	Train Accuracy	Test Accuracy
DocFormer	84.6%	61.04%
LayoutLM	58.95%	57.4%
CNNs	59.5%	60.5%

valid/loss =2.290, valid/acc =0.707, valid/precision =0.707, valid/recall =0.707, valid/f1 =0.707, train/loss =2.800, train/acc =0.682]

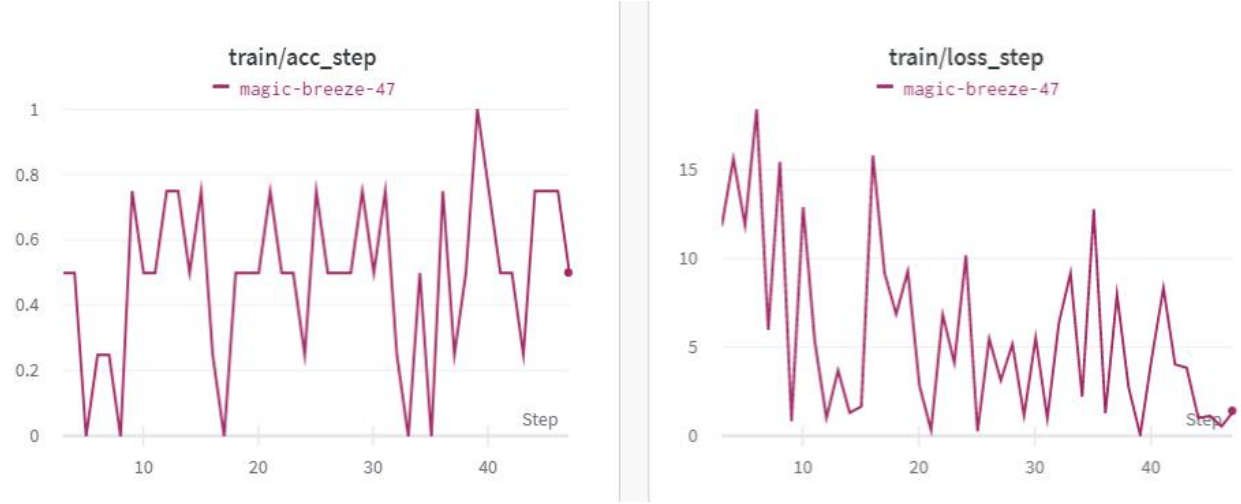
ROC-AUC Curve:



Confusion Matrix:



Deep Learning – Lab Project Report



CNNs vs Transformer Networks for task at hand:

As we've experienced throughout the course of this project that our task of Document classification can be done easily using a CNN based network approach. However, going with a transformer network helps us capture the visual NLP part which does provide us with a better classifier. The accuracies tabulated above shouldn't be the only thing we consider while building our classifier.

Conclusion:

Since we are only feeding the network 500 x 16 = 9,000 images per class giving us an epoch time of over 8hrs using Kaggle's P100 GPU, we observe a bit of overfitting that none of the regularization methods is able to tackle. However feeding more images per class should effectively fix this problem and according to refenced research paper which is trained on 3,20,000 images, should give us an f1 score of 96% on validation.

However the model as it is performs rather well on unseen data. For all experiments, we fine-tune on the training set and report numbers on the corresponding test/validation dataset. No dataset specific hyper-parameter tuning was done. We treat this as a plus and our reported numbers could be higher if dataset specific fine-tuning was done. For all downstream tasks, we use the official provided annotations unless otherwise stated. A common theme amongst these datasets is the relatively small amount of training data.

Towards the end of the project, we have spent a fair amount working with different types of CNN architectures. Different forms of information extracting through OCRs for Visual NLP as the it enhances the performance and applicability of our classifier. Through this project we've learnt a variety of new things one of which was working with pyTorch since the later transformer models weren't in the tensorflow framework which we have used in the labs.

Overall, we can achieve an accuracy in the neighbourhood of 60% which is less than what we targeted while starting this project. One integral part of learning has been learning that literature review is also an integral part of any project as it gives us insights and a-priori knowledge before

going ahead with the project. Another hurdle in our way was that the research work conducted for the same task was with a dataset which wasn't easily available and easy to work with because of its sheer size making it even harder to replicate the results from the different research papers. This adds to our skillset yet another useful and valuable skill, from report writing to literature review to pipelining all the way up-to deployment. We had placed our bets on the DocFormer model to give us a good training accuracy, which it did, which again proves that having done good literature review goes a long way.

Another point to add here is that the task domain is still very much in research with the latest findings being published on 22nd October 2022, which was the working timeline of this project so being able to take such a task and make a useful working product out of it does make a valuable experience for us. All in all, moving past those hurdles we were able to successfully accomplish our goals.

Future Scope:

Given the appropriate resources necessary to the training process, we can further train this on the bigger original RVL-CDIP dataset.

Further proposed changes:

An upgraded docformer architecture which splits the images into Header, Footer and Body(left and right) and feed the derived visual features to the transformer layer instead of the whole image as seen in docformer architecture. In preliminary testing, this method offered a boost of over 8% in

accuracy however integrating it to docformer would take a better GPU and CPU to train over a feasible time period. Further we can test the same proposed architecture with one of these components randomly masked off during training.