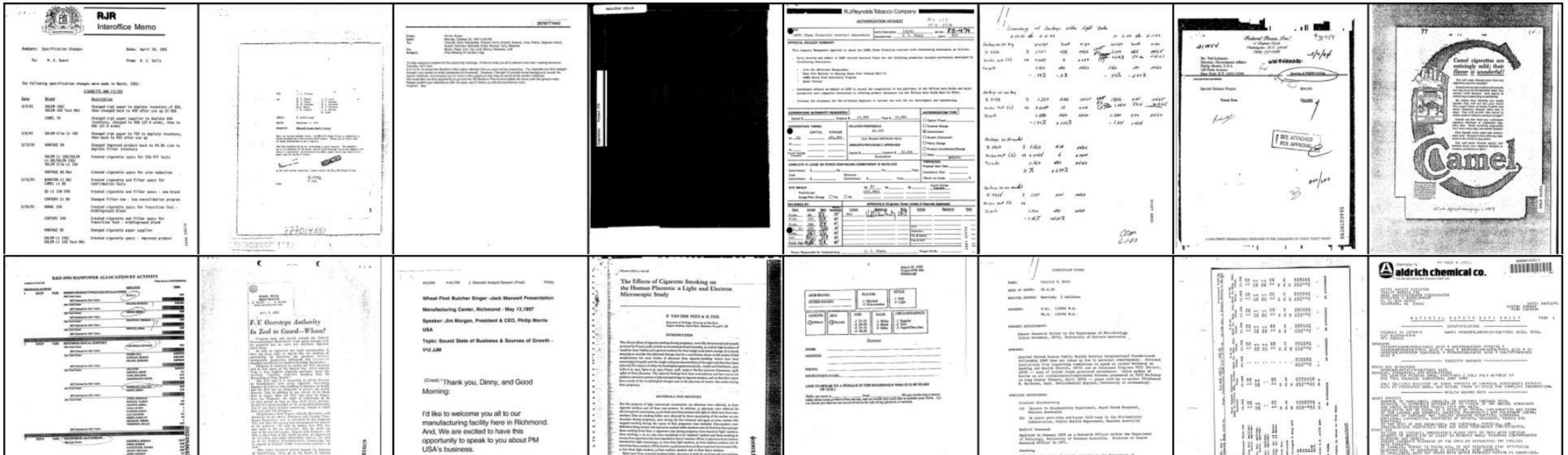
Problem Statementm, Goals and Results

A document may be a text document or it may a scanned document

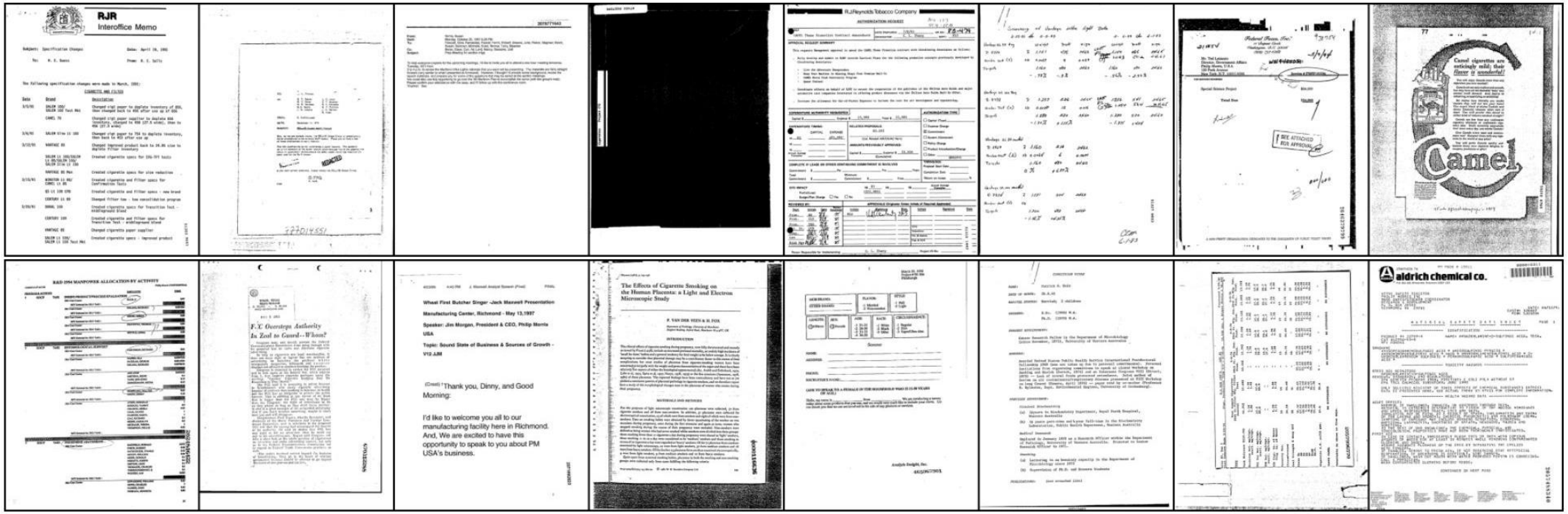
The proposed model should be able to extract all features and demonstrate the accuracy



Literature Review

- Computer Vision isn’t nearly as smart as Machine Learning
- While using Image morphological operations alone serve to extract features they do not perform as well
- Tesseract can be used to adopt a machine learning approach to train a model on image features and extract them
- It is an optical character recognition engine with open-source code, this is the most popular and qualitative OCR-library. OCR uses artificial intelligence for text search and its recognition on images. Tesseract is finding templates in pixels, letters, words and sentences.

Proposed Methodology



Proposed Methodology

- Rather than training a model on images of tables, figures and label it would be more intelligent to train it on a group of features with high correlation
- E.g. Since tables, figures and two column paragraphs are found together we identify an image containing all these features as a research paper and consider it a class
- Then the probability is predicted for the document belonging to that class. Same applies to all other classes forming a total of 16.
- Thus we calculate our model accuracy

Data

- RVL-CDIP dataset contains 40,000 images but as this is a semester project so there was limited time and no free GPU :/
- So only 800 were used
- 640 images for training (80%)
- 128 images for validation (16%)
- 32 images for testing (4%)

Results and Comparisions

- The classification model was run for 9 epochs
- Reached minimum loss of 3.4
- Training Accuracy 96.4%
- Testing Accuracy 78.1%

Conclusion and Future Works

- Addition of more classes
- More test cases

Summary

Methodology: A supervised learning classification model with 16 classes each belonging to a group of features occasionally found together.

```
[ ]  Epoch: 1
     Loss: 38.15239723920822
     Training accuracy: 58.125
     Epoch: 2
     Loss: 23.99459100961685
     Training accuracy: 74.84375
     Epoch: 3
     Loss: 13.594086346030235
     Training accuracy: 86.09375
     Epoch: 4
     Loss: 8.680811493098735
     Training accuracy: 92.5
     Epoch: 5
     Loss: 6.803154343366623
     Training accuracy: 93.4375
     Validation accuracy: 71.09375
     Epoch: 6
     Loss: 4.595013454928994
     Training accuracy: 96.09375
     Epoch: 7
     Loss: 2.6782825395464895
     Training accuracy: 97.5
     Epoch: 8
     Loss: 2.5178303118795156
     Training accuracy: 97.34375
     Epoch: 9
     Loss: 3.4446124441921713
     Training accuracy: 96.40625
```

```
19
20  accuracy = 100 * correct / test_size
21  print("Testing accuracy:", accuracy.item())

Testing accuracy: 78.125
```