Deep Learning (CSL4020)

DIGITIZATION OF HAND-WRITTEN ANSWER SCRIPTS

(Final Report)

TEAM DESCRIPTION

Akriti Gupta (B21AI005)
Pre-final Year (B.Tech. Artificial Intelligence & Data Science)

Sagnik Goswami (B21AI034)
Pre-final Year (B.Tech. Artificial Intelligence & Data Science)

Tanish Pagaria (B21AI040)
Pre-final Year (B.Tech. Artificial Intelligence & Data Science)

Uppala Giridhar (B21EE072)
Pre-final Year (B.Tech. Electrical Engineering)

IIT Jodhpur Undergraduates

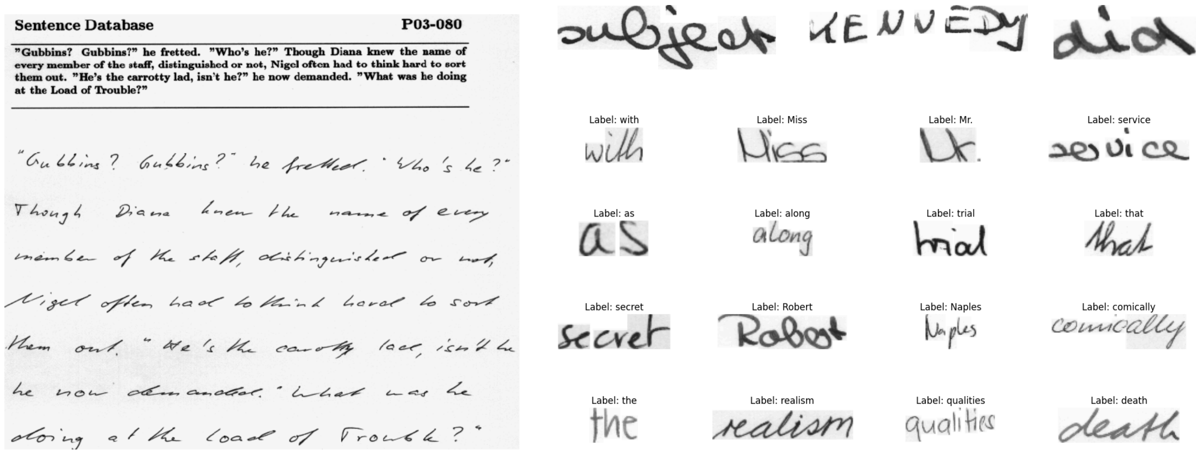
PROBLEM STATEMENT

The task was to develop a deep learning model to facilitate the digitization of handwritten answer scripts. It is a crucial task in educational settings, especially in the context of assessments, examinations, and grading. Traditional methods of grading involve manual assessment by teachers, which can be time-consuming, subjective, and prone to errors. Digitizing handwritten answer scripts through automated means offers several advantages: faster grading, consistency, and scalability.

DATASET

IAM Handwriting Database

The IAM database contains 13,353 images of handwritten lines of text created by 657 writers. The texts transcribed by these writers are from the Lancaster-Oslo/Bergen Corpus of British English.
It includes contributions from 657 writers, resulting in a total of 1,539 handwritten pages comprising 115,320 words, and it is categorized as part of the modern collection. The database is labeled at the sentence, line, and word levels.



Sentence Database **P03-080**

"Gubbins? Gubbins?" he fretted. "Who's he?" Though Diana knew the name of every member of the staff, distinguished or not, Nigel often had to think hard to sort them out. "He's the carrotty lad, isn't he?" he now demanded. "What was he doing at the Load of Trouble?"

Label: with
Label: Miss
Label: Mr.
Label: service
Label: as
Label: along
Label: trial
Label: that
Label: secret
Label: Robert
Label: Naples
Label: comically
Label: the
Label: realism
Label: qualities
Label: death

Image samples from IAM Handwriting Database

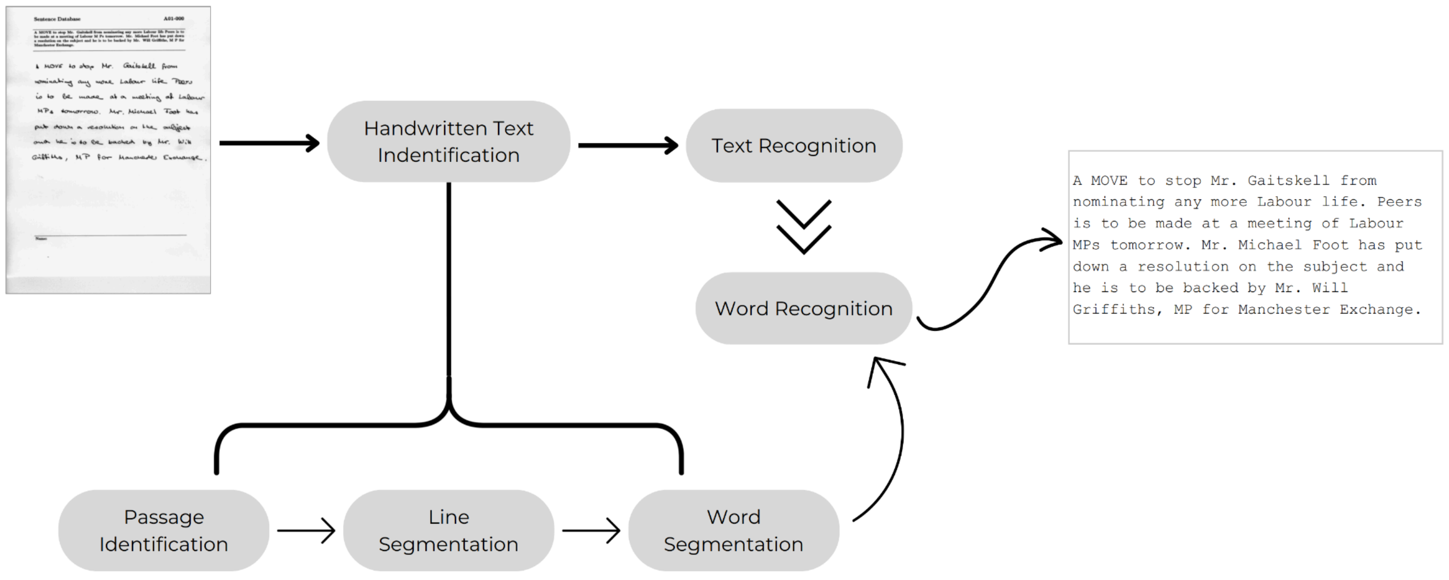
IAM Handwritten Forms Dataset

It contains complete forms of unconstrained handwritten text, which were scanned at a resolution of 300 dpi and saved as PNG images with 256 gray levels.

The IAM Handwriting Dataset was chosen because several state-of-the-art OCR models including DTrOCR, TrOCR, Self-Attention + CTC + language model, etc. have been trained on the same dataset or its subsets.

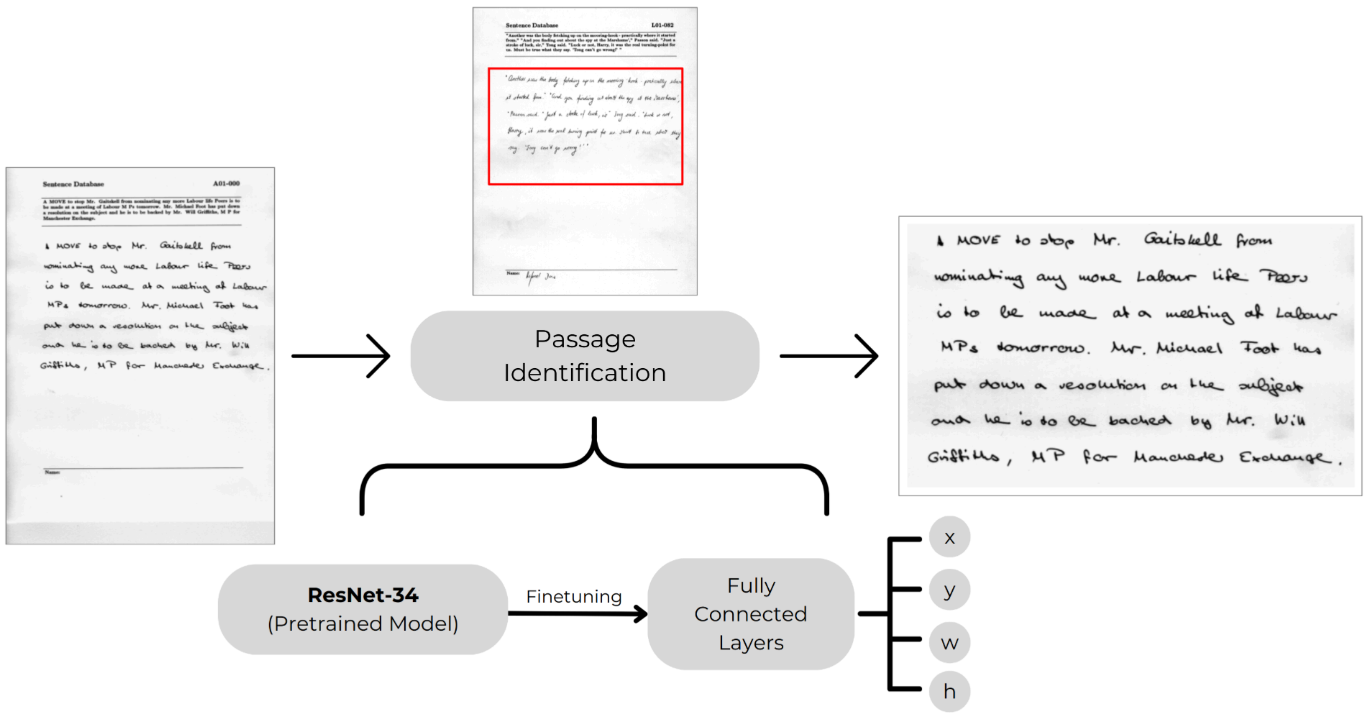
SOLUTION STRATEGY

The proposed solution strategy can be broken down as follows:



A MOVE to stop Mr. Gaitskell from nominating any more Labour life. Peers is to be made at a meeting of Labour MPs tomorrow. Mr. Michael Foot has put down a resolution on the subject and he is to be backed by Mr. Will Griffiths, MP for Manchester Exchange.

PASSAGE IDENTIFICATION



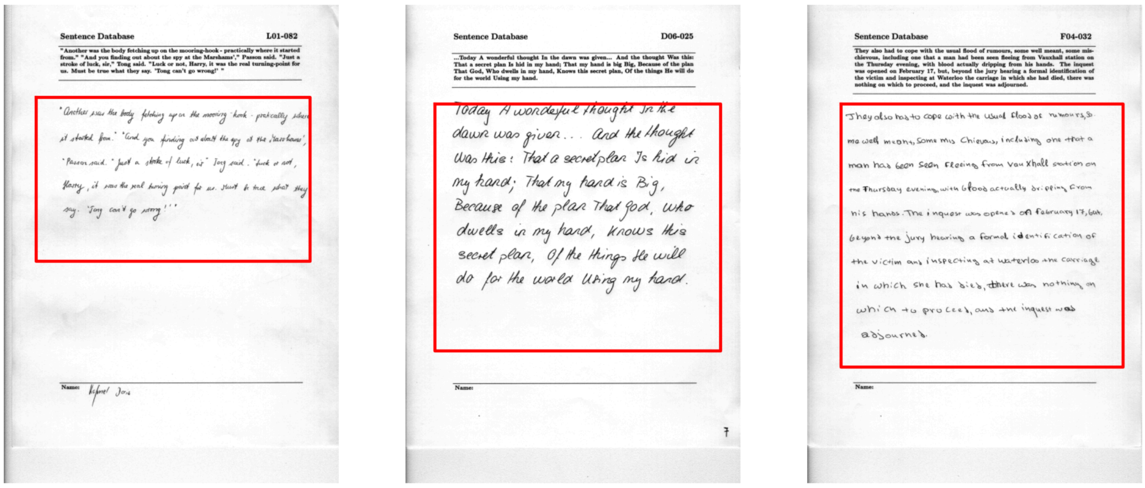
A MOVE to stop Mr. Gaitskell from nominating any more Labour life Peers is to be made at a meeting of Labour MPs tomorrow. Mr. Michael Foot has put down a resolution on the subject and he is to be backed by Mr. Will Griffiths, MP for Manchester Exchange.

It involved predicting the location of handwritten passages through bounding boxes containing *x*, *y* coordinates, width, and height, all measured in percentages relative to the page size. The approach involved **assuming the presence of one passage of printed text and one passage of handwritten text**, leveraging the IAM Handwritten Forms dataset.

The method employed a pre-trained truncated 34 layer residual network (**ResNet34**) trained on ImageNet to extract image features. The model was **fine-tuned** using bounding box data from the paragraphs in the dataset, which was generated using computer vision.

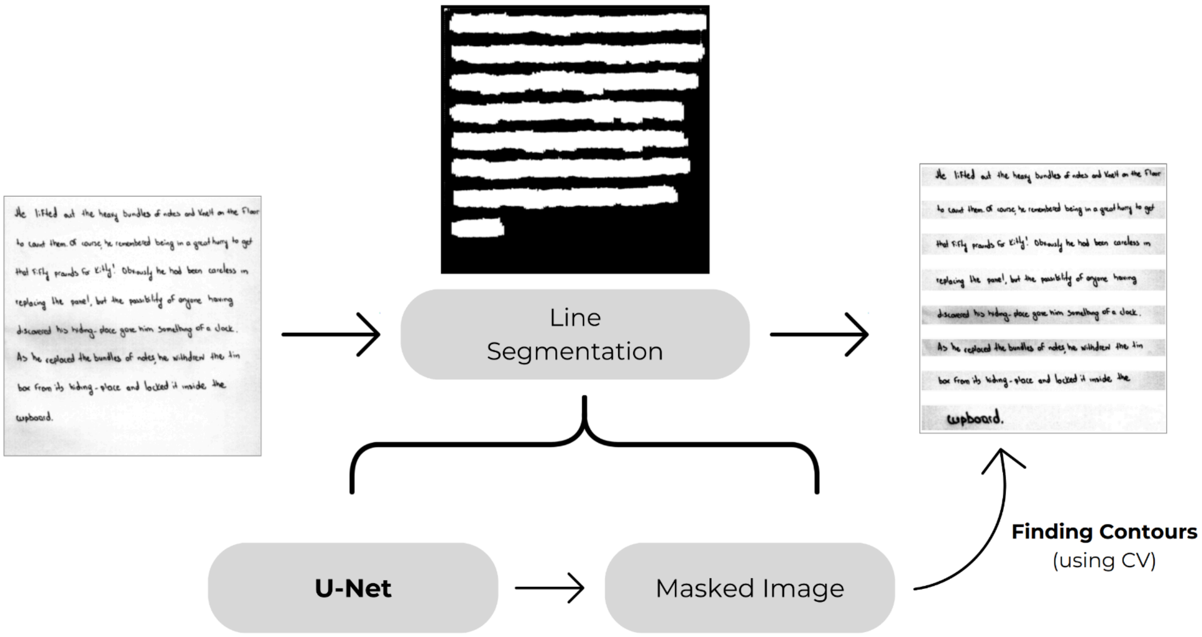
In this ResNet34 model, the first convolutional layer's weights were averaged to support grayscale images. Then, these features went through **three fully connected layers: two with 64 units each**, using **ReLU** activation, and one with **4 units using sigmoid activation**. These four units, activated by sigmoid, represented the *x*, *y* coordinates, width, and height of the bounding box, given as percentages of the page size. The network was trained to minimize the **mean squared error**.

ResNet34, a deep neural network with residual connections, excels at extracting complex image features. Its pre-training on large datasets allows for efficient fine-tuning, especially for tasks like predicting text bounding boxes. It effectively handles grayscale images and excels in capturing contextual information from a broad receptive field.

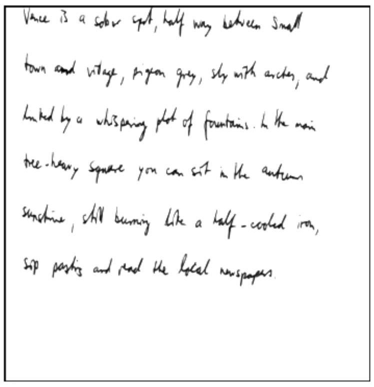


Passage identification results

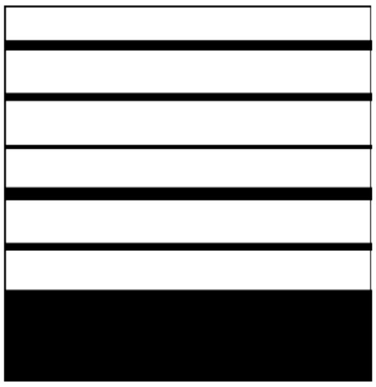
LINE SEGMENTATION



It involved segmenting the passages from the passage segmentation step into individual lines. The step involved U-Net, which was trained on the cropped passage images and the mask images (generated using computer vision) of the corresponding passage based on the lines (used as the ground truth images) from the IAM Handwritten Forms dataset.



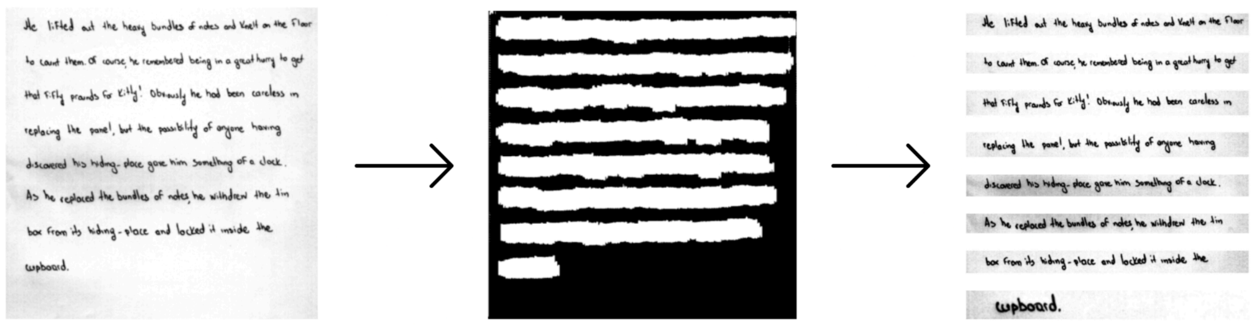
Passage image



Corresponding mask image (for training)

The U-Net architecture consisted of convolutional blocks for encoding and decoding.
The encoder comprised three blocks, each with two convolutional layers followed by max pooling, with the number of channels progressively increasing from 1 → 16 → 32 → 64. ReLU activation functions were applied after each convolutional layer.

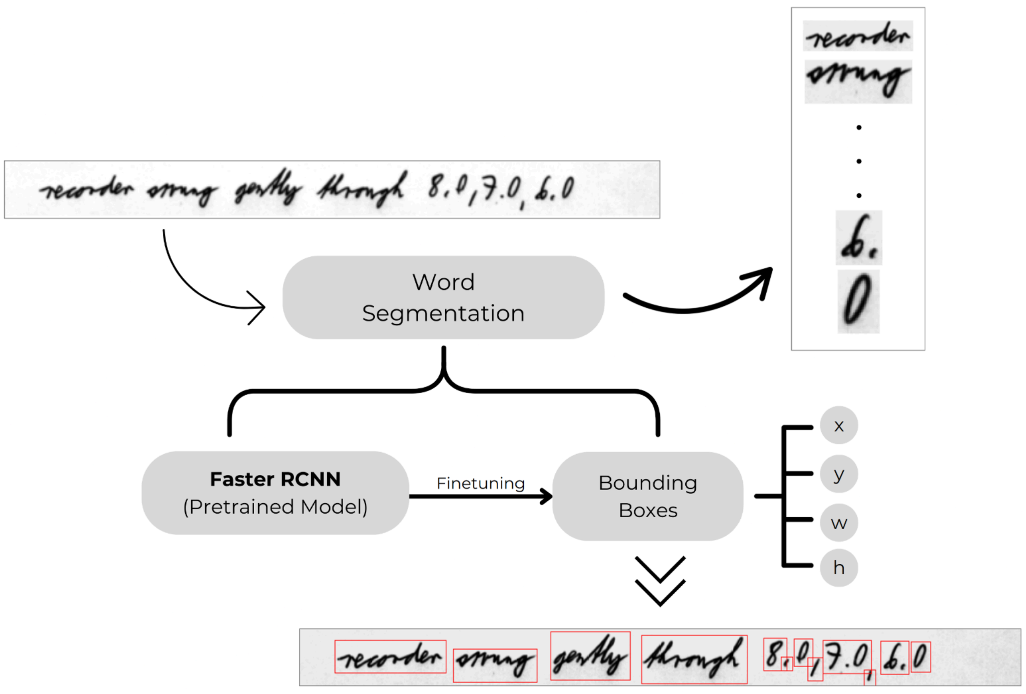
In the decoder, three blocks featured transposed convolutional layers for upsampling and concatenation with **skip connections** from the encoder. Each decoder block included two convolutional layers, ReLU activation, and dropout regularization with a rate of 0.2. The final output layer used sigmoid activation for binary classification, producing segmentation maps with two classes.
The model was trained to minimize the **categorical cross-entropy loss**.
The mask images generated from the model were used to generate lines by finding contours using computer vision.



Line segmentation results

The U-Net architecture captures global context and fine details through its encoder-decoder design. Skip connections maintain spatial information for precise localization. Dropout regularization prevents overfitting. Its segmentation capability with two classes effectively segments passages into individual lines.

WORD SEGMENTATION

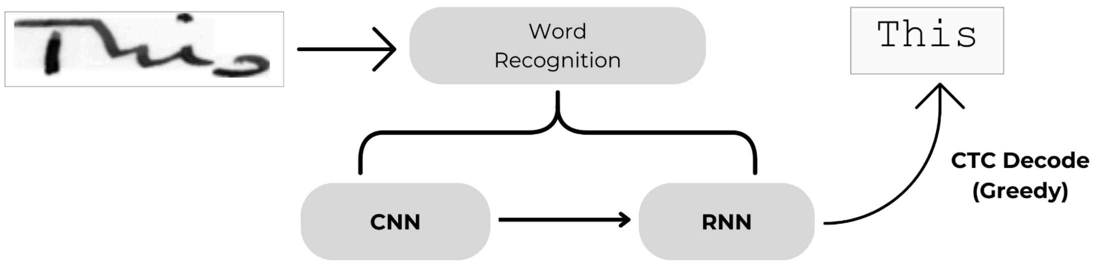


It involved segmenting a line into words. The task employed a pre-trained **Faster RCNN ResNet50 FPN** model, which was fine-tuned using bounding box data (generated using computer vision) for words corresponding to each line in the IAM Handwritten Forms dataset.
The model generated bounding box coordinates for the words in the corresponding line.

Faster RCNN is a state-of-the-art object detection framework and is highly accurate. It was used for the task because of the following key features:

- Region Proposal Network (RPN): Efficiently proposes word-containing regions, optimizing computation.
- Feature Pyramid Network (FPN): Detects words across different scales and styles, aiding segmentation in varied text images.
- ResNet50: Extracts rich semantic features, enhancing word segmentation accuracy.
- Localization and Classification: Precisely locates word bounding boxes and determines word presence within regions. Enables streamlined end-to-end training for performance optimization.

WORD RECOGNITION



It involved generating text corresponding to a word image obtained from the previous steps.
The following characters were identified from the word image and label data from the IAM Handwriting Database:
`!\"#&'()*+,-./0123456789:;?ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz`

Each word text sequence was converted into a vector of a fixed maximum length (64 in this case) by mapping the characters in the word to the corresponding index in the above character array, and the remaining vector was filled with the number 79, representing a blank character.

A (**CNN + RNN**) based model was employed for this task.
The CNN model had seven convolutional layers with varying output channels (64, 128, 256 twice, and 512 thrice), each followed by ReLU activation and batch normalization. Max-pooling was applied after the 1st, 2nd, 4th, and 6th convolutional layers, with a final 2x2 convolutional layer for feature representation.
The RNN model consisted of a **bidirectional LSTM** with two layers, having input and hidden sizes of 512 and 256 respectively, with a 0.2 dropout rate. A fully connected layer followed the LSTM for final output.
The overall model combined the CNN and RNN architectures. The input passed through the CNN to extract features, then through the RNN for sequential processing. The final output was obtained from the RNN using **CTC Greedy Decode**.

CTC (Connectionist Temporal Classification) Greedy is ideal for sequence-to-sequence tasks due to its adaptability to variable-length sequences without explicit alignment between input and output. It efficiently predicts characters individually, regardless of their image regions, making it suitable for character-level prediction. Its computational efficiency seamlessly integrates with CNN + RNN architectures, facilitating streamlined text output generation.
CTC Greedy is approx. 11 times faster than CTC Beam Search.

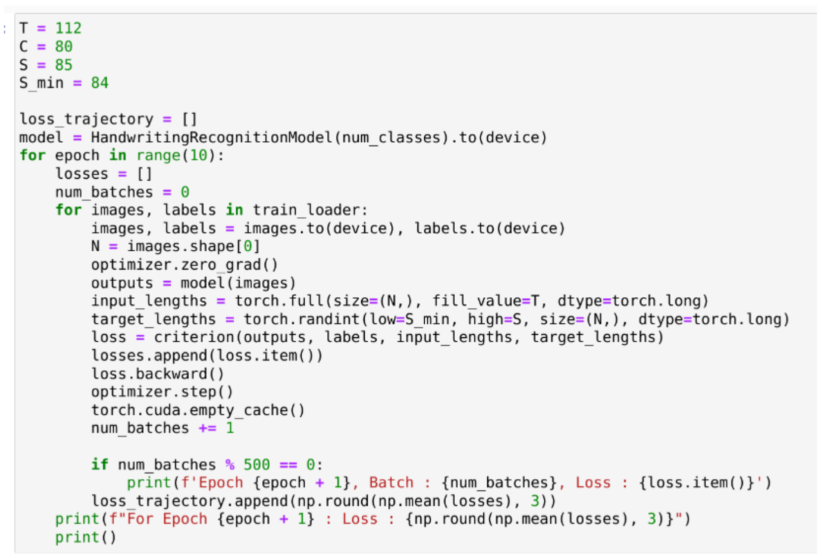
INNOVATIONS

- A simple and computationally efficient architecture was employed for the whole task. The model generated output in significantly less time as compared to the state-of-the-art architectures. Detection at word level is computationally faster and memory-efficient as compared to the state-of-the-art models which predict multiple lines together at the same time.
- U-Net architecture was employed for extracting line parts from the segmented passages.
- Despite the unavailability of the complete official IAM Handwriting Dataset required for specific tasks like segmentation and text recognition, a major portion of the datasets was generated using computer vision.
- State-of-the-art model TrOCR requires single line images for recognition. In comparison, the proposed model could take in the whole image and generate line-by-line text.
- The images were processed using computer vision techniques including gamma correction to handle contrast and brightness of the input images.

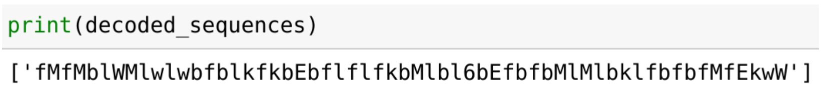
CHALLENGES

The following challenges were encountered during the project:

- An attempt was made to perform image text recognition directly on the segmented lines instead of the proposed line-to-word segmentation and then word recognition. The CTC loss metric used during the training, which worked well for the word recognition task, failed in the case of line text recognition.
  The CTC loss frequently hit infinity (displayed as zero in the image below).



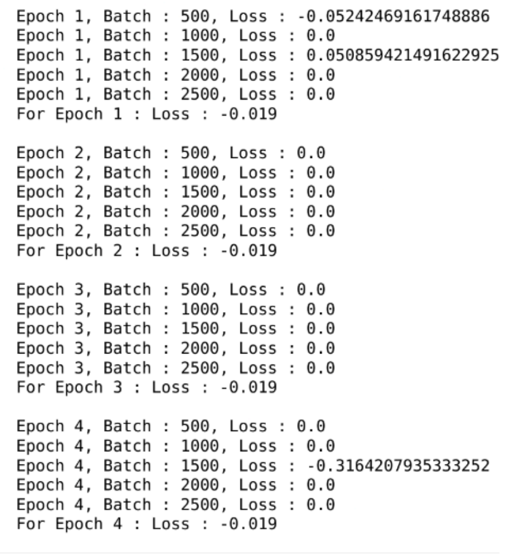
```
T = 112
C = 80
S = 85
S_min = 84

loss_trajectory = []
model = HandwritingRecognitionModel(num_classes).to(device)
for epoch in range(10):
    losses = []
    num_batches = 0
    for images, labels in train_loader:
        images, labels = images.to(device), labels.to(device)
        N = images.shape[0]
        optimizer.zero_grad()
        outputs = model(images)
        input_lengths = torch.full(size=(N,), fill_value=T, dtype=torch.long)
        target_lengths = torch.randint(low=S_min, high=S, size=(N,), dtype=torch.long)
        loss = criterion(outputs, labels, input_lengths, target_lengths)
        losses.append(loss.item())
        loss.backward()
        optimizer.step()
        torch.cuda.empty_cache()
        num_batches += 1

        if num_batches % 500 == 0:
            print(f'Epoch {epoch + 1}, Batch : {num_batches}, Loss : {loss.item()}')
        loss_trajectory.append(np.round(np.mean(losses), 3))
    print(f"For Epoch {epoch + 1} : Loss : {np.round(np.mean(losses), 3)}")
    print()
```



```
print(decoded_sequences)
```

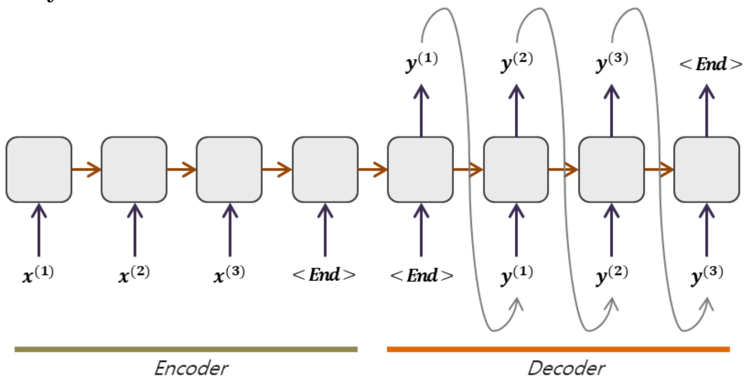
```
['fMfMblWMlwlwbfblkfkbEbflflfkbMlbl6bEfbfbMlMlbklfbfbfMfEkwW']
```

- An attempt was made to implement a spelling checker for the recognized text to improve the overall result. A sequence-to-sequence (**Seq2Seq**) model architecture with an encoder-decoder structure utilizing Long Short-Term Memory (LSTM) cells was employed for this task.



```
Epoch 1, Batch : 500, Loss : -0.05242469161748886
Epoch 1, Batch : 1000, Loss : 0.0
Epoch 1, Batch : 1500, Loss : 0.050859421491622925
Epoch 1, Batch : 2000, Loss : 0.0
Epoch 1, Batch : 2500, Loss : 0.0
For Epoch 1 : Loss : -0.019

Epoch 2, Batch : 500, Loss : 0.0
Epoch 2, Batch : 1000, Loss : 0.0
Epoch 2, Batch : 1500, Loss : 0.0
Epoch 2, Batch : 2000, Loss : 0.0
Epoch 2, Batch : 2500, Loss : 0.0
For Epoch 2 : Loss : -0.019

Epoch 3, Batch : 500, Loss : 0.0
Epoch 3, Batch : 1000, Loss : 0.0
Epoch 3, Batch : 1500, Loss : 0.0
Epoch 3, Batch : 2000, Loss : 0.0
Epoch 3, Batch : 2500, Loss : 0.0
For Epoch 3 : Loss : -0.019

Epoch 4, Batch : 500, Loss : 0.0
Epoch 4, Batch : 1000, Loss : 0.0
Epoch 4, Batch : 1500, Loss : -0.3164207935333252
Epoch 4, Batch : 2000, Loss : 0.0
Epoch 4, Batch : 2500, Loss : 0.0
For Epoch 4 : Loss : -0.019
```

However, because of the unavailability of a good dataset, noise was added to a given English language dataset, which did not perform well. Also, a limited subset of the dataset could be used at a time because of memory limitations.



Seq2Seq architecture

RESULTS

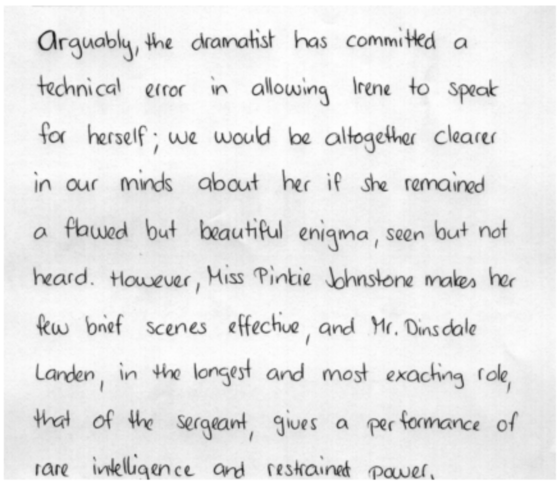
The lines dataset from the IAM Handwriting Dataset testing portion was used as the test set for the word recognizer model. The performance metric used was **Character Error Rate (CER)**.
The Character Error is the percentage of characters that have been transcribed incorrectly by a text recognition model.

$$CharErrorRate = \frac{S+D+I}{N} = \frac{S+D+I}{S+D+C}$$

where:

S is the number of substitutions,
D is the number of deletions,
I is the number of insertions,
C is the number of correct characters,
N is the number of characters in the reference ($N = S + D + C$).

CER obtained for 1000 test images: `0.3449564278125763`.



Arguably, the dramatist has committed a
technical error in allowing Irene to speak
for herself; we would be altogether clearer
in our minds about her if she remained
a flawed but beautiful enigma, seen but not
heard. However, Miss Pinkie Johnstone makes her
few brief scenes effective, and Mr. Dinsdale
Landen, in the longest and most exacting role,
that of the sergeant, gives a performance of
rare intelligence and restrained power.

```
orguably the oeanatist has committed a
teonnica euor in ellowing tree too speak
for herssl wer would be altoghe cleore
po.n uor minds abool her if she reniieed pr.
a faaned bont beuati engina seen bont not
heard Houeer Miss Grinc buntere make her
ter loice scenes effeciie and Me. por lindde
Lander Me. in the longes and most exerting rale
shett iot the Seggent " gives sa parance wole
mare intinoue and retaan re puer
```

Model predictions for a random sample image from the test set

SOLUTION ANALYSIS & POSSIBLE WEAKNESSES

- The proposed solution works well for images containing only text, and gives decent results, consuming less time and memory.
- The recognition at word-level improves computational efficiency in comparison to architectures identifying multiple text lines simultaneously.
- The U-Net model for line segmentation has a simpler architecture compared to object detection algorithms like Single Shot Multibox Detector (SSD), still giving comparable results.
- The use of pretrained ResNet34 model trained on ImageNet dataset extracts rich and salient features from the image very efficiently.
  The bounding box coordinate predictor (x, y, w, h) modeled as a regression problem gives comparable results to object detection algorithms, taking less time and space.

The possible weaknesses for the proposed solution may include:

- Instead of CTC Greedy Decode used in the proposed solution, a language model denoiser with beam search could have been used which would have proved effective for vocabulary words like names and places. But the proposed solution did not use any Transformer-based architecture, which might have improved the overall performance.
- The model may produce inaccurate or unreliable results in the presence of noise or variability. One possible scenario may be when the page is not properly segmented from the background. An example case is shown below:

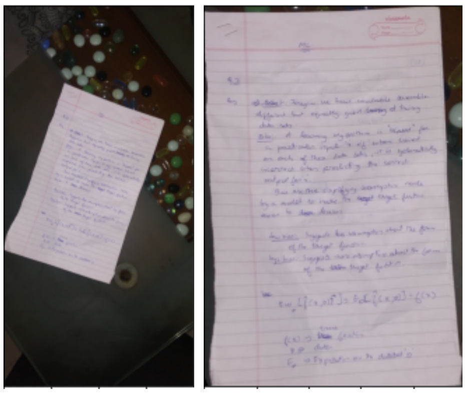


Image Sample **vs** *Expected Image*

A possible proposed solution for the above is given by LearnOpenCV.

- The IAM Handwriting Dataset was based on only english language characters, so the proposed solution cannot be employed for mathematical equations, diagrams or text in any other language.
- The proposed solution could not detect the text at line level as a whole, which could have been helpful in identifying contextual information about the previous, current and next words, thus improving the predictions.

CONCLUSION

The proposed solution attempted to identify and digitize text in handwritten answer scripts. There is always room for improvement; humans also occasionally struggle with recognizing handwritten texts, a challenge the proposed solution successfully addressed in many cases. The next step toward a better approach could involve utilizing a larger and more diverse dataset, as well as exploring and enhancing state-of-the-art methods.

REFERENCES

Prominent past work in handwritten text recognition
https://paperswithcode.com/task/handwritten-text-recognition
http://vision.stanford.edu/teaching/cs231n/reports/2017/pdfs/810.pdf

DTrOCR, TrOCR & S-Attn/CTC **(Literature Survey)**
https://paperswithcode.com/paper/dtrocr-decoder-only-transformer-for-optical
https://arxiv.org/pdf/2308.15996v1.pdf
https://paperswithcode.com/paper/trocr-transformer-based-optical-character
https://utorontomist.medium.com/handwriting-recognition-using-deep-learning-14ec078872b0
https://arxiv.org/pdf/2104.07787v2.pdf

Faster RCNN
https://medium.com/alegion/faster-r-cnn-using-region-proposal-network-for-object-detection-c113b6dd00a
https://debuggercafe.com/object-detection-using-pytorch-faster-rcnn-resnet50-fpn-v2/
https://link.springer.com/chapter/10.1007/978-981-99-8451-0_24

ResNet34
https://medium.com/nerd-for-tech/handwriting-recognition-more-than-just-mnist-36e68832de73

U-Net
https://www.researchgate.net/publication/339025343_Text_Line_Segmentation_in_Historical_Document_Images_Using_an_Adaptive_U-Net_Architecture

CER
https://help.transkribus.org/character-error-rate-and-learning-curve
https://lightning.ai/docs/torchmetrics/stable/text/char_error_rate.html

More on evaluation methods, techniques, evaluation metrics, etc.
https://nanonets.com/blog/handwritten-character-recognition/
https://keras.io/examples/vision/handwriting_recognition/