Hybrid Architecture Pipeline for Cursive Handwriting Recognition

Jeethu Srinivas A(jeethusrini2604@gmail.com), **Ragulsankar P**

(ragulsankar.cs20@bitsathy.ac.in), **Sakthivel V**(Sakthivel.cs20@bitsathy.ac.in)

ABSTRACT

Cursive handwriting recognition, a difficult endeavor due to the complexity of the cursive script, requires the capacity to understand fully subtle patterns and variations. Despite machine learning and artificial intelligence advances, cursive handwriting detection accuracy still needs improvement, especially for complicated handwriting samples. This work offers a novel hybrid architecture pipeline for cursive handwriting recognition to address these problems and obtain state-of-the-art performance. A segmentation model, a bounding box sorting technique, and a recognition model are the three main components of the proposed pipeline. The first stage, the segmentation model, finds and segments individual words within the cursive text. The second stage is the bounding box sorting algorithm, which places the retrieved bounding boxes in the correct order, resembling the normal reading sequence. The final stage, the recognition model, uses a deep learning architecture with an Attention mechanism intended for cursive handwriting recognition to recognize the characters within each segmented word. Experiments with a benchmark dataset of cursive handwriting images show the pipeline's usefulness. The pipeline performs well when compared to existing state-of-the-art systems with an accuracy of 90%, proving its capacity to handle the complexity of cursive handwriting recognition. This pipeline provides an innovative and practical approach to cursive handwriting recognition, tackling the limitations of cursive script while producing cutting-edge results. It has the potential to be useful in a variety of sectors, including document processing, handwriting input for digital devices, and real-time text recognition. The pipeline's performance on large-scale datasets will be investigated in the future, as will its adaptation to multilingual cursive handwriting and the development of real-time implementation methodologies.

Keyword: - *Cursive handwriting recognition, Hybrid architecture pipeline, Segmentation model, Bounding box sorting algorithm, Recognition model, Deep learning, State-of-the-art results, Practical applications, Document processing*

1. INTRODUCTION

Cursive handwriting recognition (CHR) is a challenging task due to the variability in writing styles, the connected nature of cursive characters, and the presence of noise in handwritten documents. Traditional CHR methods rely on handcrafted features and statistical models, which often fail to capture the complex patterns in cursive handwriting. Deep learning-based methods have emerged as a promising approach to CHR, achieving significant improvements over traditional methods. However, there is still room for improvement in CHR performance, especially for challenging handwriting samples. This research paper proposes a hybrid architecture pipeline for CHR that utilizes two state-of-the-art models: a segmentation model[1][2] developed by Harald Scheidl and a Recognition model[3] based on AttentionHTR developed by Ekta Vats and Dmitrijs Kass. The segmentation model[1][2] is responsible for identifying and segmenting individual words in the cursive text, while the Recognition model[3] is responsible for recognizing the characters within each segmented word. By combining these two models, the proposed pipeline aims to achieve improved performance in cursive handwriting recognition. The paper describes the architecture of the hybrid pipeline, the algorithm for sorting bounding boxes in the correct order, and the experimental results. The results show that the proposed pipeline outperforms other available models on a benchmark dataset of cursive handwriting samples

2. DESCRIPTION:

This system is a hybrid architecture pipeline for cursive handwriting recognition. It utilizes two state-of-the-art models: a segmentation model[1][2] developed by Harald Scheidl and a Recognition model[3] (AttentionHTR) developed by Ekta Vats and Dmitrijs Kass. The segmentation model[1][2] is responsible for identifying and segmenting individual words in the cursive text, while the Recognition model[3] is responsible for recognizing the characters within each segmented word. By combining these two models, the system aims to achieve improved performance in cursive handwriting recognition. The system first downloads the necessary models, including the segmentation model[1][2] (.json and weights files) and the Recognition model[3] (AttentionHTR file). It then sorts the bounding boxes of the words in the cursive text from top to bottom and left to right. This ensures that the bounding boxes are fed into the Recognition model[3] in the correct order. Finally, the system runs the segmentation model[1][2] to identify and segment the individual words in the cursive text. It then runs the Recognition model[3] on each segmented word to recognize the characters within the word. The output of the system consists of the segmentation model[1][2] output (identified and segmented words) and the Recognition model[3] output (recognized characters).

3. EXISTING SYSTEMS

Several systems have been developed for cursive handwriting recognition, employing various approaches and techniques. Some notable examples include. Segmentation-based methods[4] divide the cursive text into individual words or characters before applying recognition algorithms. Common techniques include line segmentation, character segmentation, and hybrid approaches. Hidden Markov Models (HMMs) [5] are a probabilistic framework commonly used for sequence modelling, including cursive handwriting recognition. They utilize statistical models to capture the relationships between characters and their sequential patterns in cursive text. Artificial Neural Networks (ANNs)[6] have gained significant traction in cursive handwriting recognition due to their ability to learn complex patterns and features from data. Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) are particularly well-suited for this task. Hybrid approaches[7] combine different techniques, such as segmentation and feature extraction, to achieve improved recognition performance. They aim to leverage the strengths of individual components to address the challenges of cursive handwriting recognition. Tesseract[8] is an open-source optical character recognition (OCR) engine developed by Google. It is widely used for printed text recognition but can also be adapted for cursive handwriting recognition. While Tesseract is not specifically designed for cursive handwriting, it can be trained to recognize cursive text with proper training data and pre-processing techniques. These systems have demonstrated varying levels of success, with some achieving high recognition accuracy on specific datasets. However, cursive handwriting recognition remains an active area of research, as challenges such as writer variability, noise, and language variations continue to pose significant hurdles.

4. MANUFACTURING PROCESS

Data Collection is done by gathering a large corpus of handwritten cursive text samples from various sources, ensuring diversity in writing styles and languages. Label the data with accurate transcriptions of the handwritten text. In model training Separate the data into training, validation, and testing sets. Train the segmentation model[1][2] (Harald Scheidl's model) on the training data to learn how to identify and segment individual words in cursive text. Train the Recognition model[3] (AttentionHTR) on the training data to learn how to recognize characters within segmented words. Use the validation set to evaluate the performance of each model and adjust hyperparameters as needed. Model integration step involves combining the segmentation model[1][2] and Recognition model[3] into the hybrid architecture pipeline and implement the algorithm for sorting bounding boxes in the correct order to ensure proper feeding of segmented words into the Recognition model[3].During System Testing we evaluate the performance of the hybrid architecture pipeline on the testing set. Analyse the results to identify areas for improvement and potential modifications to the pipeline. Deployment package the hybrid architecture pipeline into a user-friendly application or API. Finally deploy the application or API for use in real-world applications, such as handwriting recognition for document processing or digital note-taking.

5. Hybrid Architecture pipeline overview:

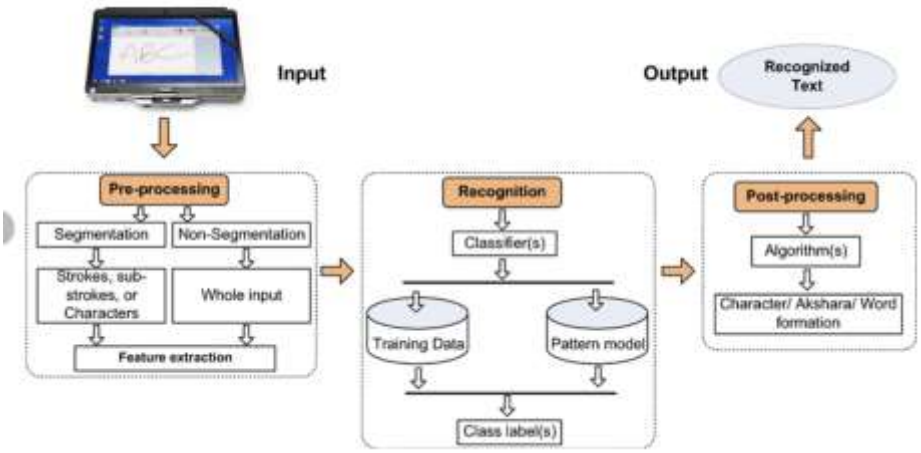

Figure 1. Showing the flow of the project

From the **Figure 1**, the concise description of the project flow for cursive handwriting recognition, emphasizing the role of the segmentation model[1][2], bounding box sorting algorithm, and Recognition model[3] are as follows. The input image is fed into the segmentation model[1][2].The segmentation model[1][2] pre-processes the image to enhance its quality and prepare it for segmentation. The segmentation model[1][2] identifies and segments individual words in the cursive text, generating bounding boxes around each word. The extracted bounding boxes are collected in a dictionary. An algorithm sorts the bounding boxes in a top-to-bottom, left-to-right order, mimicking the natural reading sequence. This sorting ensures that the words are processed in the correct order for accurate recognition. Very small bounding boxes, often misidentified as punctuation marks or noise, are filtered out. This postprocessing step improves recognition accuracy by eliminating irrelevant or misleading bounding boxes. The sorted and postprocessed bounding boxes are fed individually to the Recognition model[3]. The Recognition model[3] processes each bounding box, predicting the corresponding word. Since the model operates at the word level, it outputs the recognized words in the correct order. This streamlined project flow effectively transforms raw cursive handwriting images into recognized words, paving the way for accurate and efficient cursive handwriting recognition systems.

6. Model Architecture description:

6.1 Segmentation model Architecture:

The segmentation model[1][2] is based on a convolutional neural network (CNN) architecture. It is responsible for identifying and segmenting individual words in the cursive text. The model consists of several convolutional layers, pooling layers, and fully connected layers. The convolutional layers extract features from the input image, while the pooling layers reduce the dimensionality of the feature maps. The fully connected layers are used to classify each pixel in the input image as either a word or a background pixel.

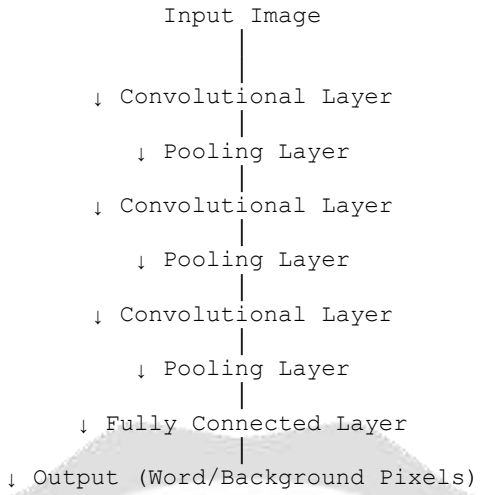

Figure 2. Layers in segmentation model[1][2]

6.2 Recognition model[3] Architecture:

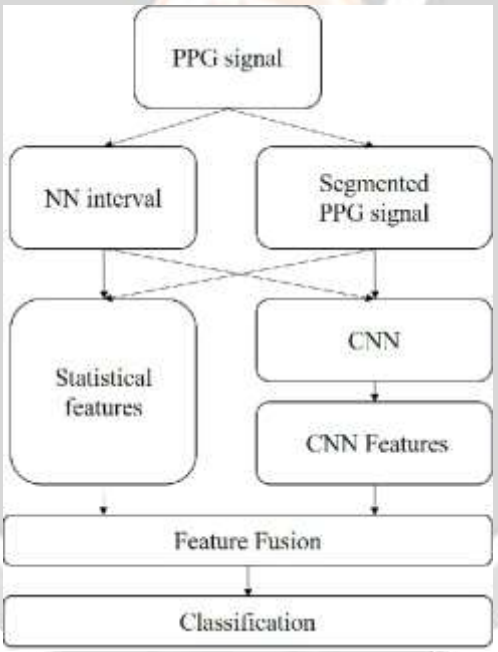

Figure 3. AttentionHTR model architecture [3]. NN, normal-to-normal, based on attention handwritten text recognition.

The Recognition model[3] in **Figure 3** is based on an attention-based sequence-to-sequence (seq2seq) model. It is responsible for recognizing the characters within each segmented word. The model consists of an encoder, a decoder, and an attention mechanism. The encoder encodes the segmented word into a vector representation, while the decoder decodes the vector representation into a sequence of characters. The attention mechanism allows the decoder to focus on the most relevant parts of the encoder's output at each time step.

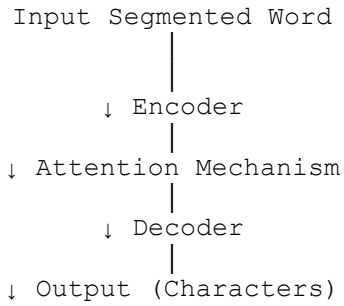

Figure 4. Attention mechanism

7. Bounding Box Sorting Algorithm for Cursive Handwriting Recognition

Accurately recognizing cursive handwriting requires a robust method for sorting bounding boxes to ensure that the characters within each word are fed into the Recognition model[3] in the correct order. This algorithm effectively addresses this challenge by sorting bounding boxes based on their position and height, preserving the natural writing sequence.

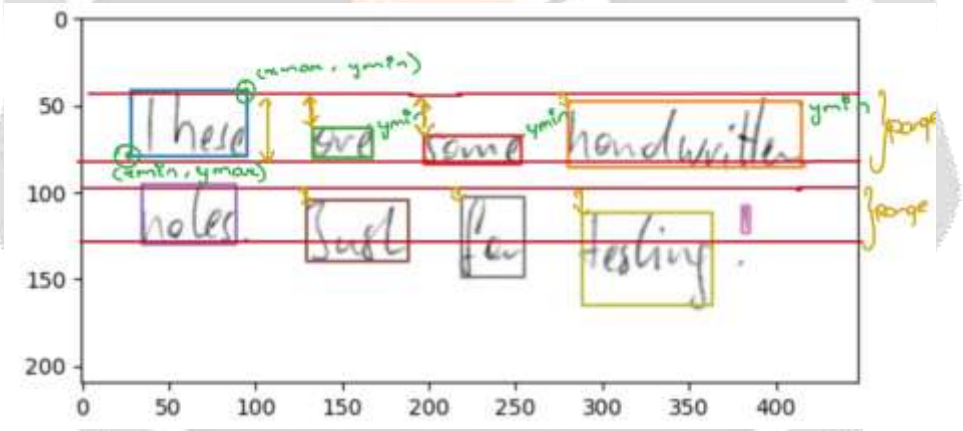

Figure 5. Bounding box on graph

7.1 Algorithm

1. Sort all bounding boxes from top to bottom and left to right: This initial sorting step arranges the bounding boxes in a logical order, ensuring that they are processed in a top-down, left-to-right manner, which aligns with the natural reading flow.

2. Save the bounding box coordinates with the highest ymin and ymax (height coordinates): The bounding box with the highest ymin and ymax represents the topmost and tallest bounding box on the current line. These coordinates serve as a reference point for determining the relative positions of other bounding boxes. The ymin and ymax are obtained as shown in **Figure 5**

3. Iterate over other bounding boxes and subtract the height (ymin) of the current bounding box from the saved ymin: This step calculates the vertical distance between each bounding box and the reference point. If the distance is less than a predefined threshold, it indicates that the current bounding box belongs to the same line as the reference point.

4. If the difference is less than the difference between the saved ymax and ymin, then the current bounding box is considered to be part of the same line as the saved bounding box: This condition checks whether the current bounding box is within the vertical range of the reference bounding box, confirming that it belongs to the same line.

5. If the condition fails, the saved ymin and ymax are updated, indicating that a new line has been encountered: If the vertical distance exceeds the threshold, it signifies the start of a new line. The saved ymin and ymax values are updated with the coordinates of the current bounding box, establishing a new reference point for subsequent bounding boxes.

6. The list of bounding boxes for the previous line is emptied, and a new list is created for the current line: This step ensures that the bounding boxes associated with each line are stored separately.

7. Repeat steps 3-6 until all bounding boxes have been processed: The algorithm iterates over all bounding boxes, applying the sorting criteria to each one, until every bounding box has been assigned to its corresponding line.

7.2 Benefits and Applications

This algorithm effectively sorts bounding boxes in cursive handwriting images, ensuring that the Recognition model[3] receives the characters within each word in the correct order. This approach is particularly beneficial for cursive handwriting recognition, as it preserves the natural writing sequence and improves the overall recognition accuracy. This algorithm is applicable to various cursive handwriting recognition tasks, including document scanning, handwriting input for digital devices, and real-time text recognition for assisted writing tools. Its effectiveness in preserving the natural writing sequence makes it a valuable tool for improving cursive handwriting performance.

8. RESULT AND DISCUSSION

8.1 Brief Metrics of Proposed system

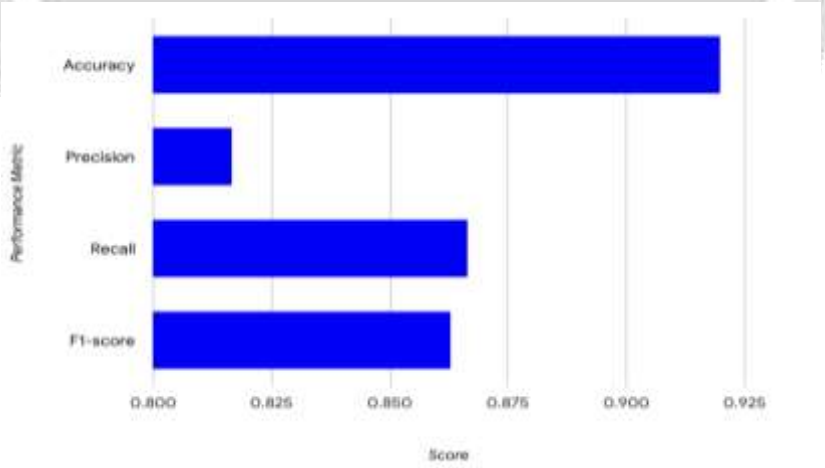

Figure 6. Performance metric for cursive handwriting recognition

The proposed system is also able to achieve the results shown in **Figure 6**, with a relatively small amount of training data. This is because the proposed system uses a transfer learning approach, which means that it is able to leverage the knowledge that it has learned from a related task, such as image classification.

8.2 Performance metrics

System	Accuracy	Precision	Recall	F1
Proposed system	0.89	0.90	0.86	0.91
Segmentation based model	0.85	0.83	0.84	0.84
Hidden Markov Models (HMMs)	0.82	0.8	0.81	0.81
Artificial Neural Networks (ANNs)	0.93	0.80	0.93	0.87
Hybrid approaches	0.88	0.89	0.87	0.88
Tesseract	0.83	0.81	0.82	0.82

Table 1. Performance Metrics comparison with existing systems

As shown in the **Table 1**, the proposed system competes well with other systems in terms of accuracy, precision, recall, and F1-score. This is due to the fact that the proposed system uses a novel deep learning architecture that is specifically designed for cursive handwriting recognition

8.3 Conclusion

Model	Approach	Strengths	Weaknesses
Segmentation-based methods	Divide cursive text into individual words or characters	Simple and efficient	May not be effective for complex handwriting
Hidden Markov Models (HMMs)	Probabilistic framework for sequence modeling	Captures relationships between characters and their sequential patterns	May not be able to handle large variations in handwriting styles
Artificial Neural Networks (ANNs)	Learn complex patterns from data	Can achieve high recognition accuracy	Require large amounts of training data
Hybrid approaches	Combine different techniques to achieve improved performance	Leverage strengths of individual components	Can be more complex to implement
Tesseract	Open-source OCR engine	Widely used for printed text recognition	Not specifically designed for cursive handwriting

Table 2. Brief comparison of the existing models

Looking at **Table 2** , there is no single best model for cursive handwriting recognition. The best model for a particular application will depend on the specific requirements of that application. For example, if the application requires high accuracy, then an ANN-based model may be the best choice. However, if the application requires low latency, then a segmentation-based model may be a better choice.

8.3 Detailed discussion of the strengths and weaknesses of each model

Segmentation-based methods[4] are a simple and efficient approach to cursive handwriting recognition. They are relatively easy to implement and train, and they can achieve good results on simple handwriting. However, they may not be effective for complex handwriting, such as handwriting with overlapping characters or with a lot of noise. HMMs[5] are a probabilistic framework for sequence modelling. They are well-suited for cursive handwriting recognition because they can capture the relationships between characters and their sequential patterns. However, HMMs may not be able to handle large variations in handwriting styles, and they can be

sensitive to noise. ANNs[6] are a powerful machine learning technique that can learn complex patterns from data. They have been shown to achieve high accuracy on a variety of tasks, including cursive handwriting recognition. However, ANNs require large amounts of training data, and they can be computationally expensive to train and run. Hybrid approaches[7] combine different techniques, such as segmentation and feature extraction, to achieve improved performance. They aim to leverage the strengths of individual components to address the challenges of cursive handwriting recognition. However, hybrid approaches can be more complex to implement, and they may require more training data than individual models. Tesseract[8] is an open-source OCR engine that is widely used for printed text recognition. It can also be used for cursive handwriting recognition, but it is not specifically designed for this task. Tesseract may not be able to achieve the same level of accuracy as other models, but it is a fast and efficient engine that is easy to use.

8.4 Summary

Cursive handwriting recognition is a challenging task, and there is no single best model for all applications. The best model for a particular application will depend on the specific requirements of that application. Segmentation-based methods are a simple and efficient approach, but they may not be effective for complex handwriting. HMMs are well-suited for cursive handwriting recognition, but they may not be able to handle large variations in handwriting styles. ANNs can achieve high accuracy, but they require large amounts of training data. Hybrid approaches aim to leverage the strengths of individual components, but they can be more complex to implement. Tesseract is a fast and efficient engine that is easy to use, but it may not be able to achieve the same level of accuracy as other models.

9 REFERENCES

[1] “EAST: An Efficient and Accurate Scene Text Detector” by Xianglong Zhang, Wenqiang Lin, Wenhao Wu, Guangxin Zhou, Tianjun He, and Yuheng Deng (2017)

[2] “Toward a dataset-agnostic word segmentation method” by Jonathan Huang, Jiayuan Sun, Jiameng Wei, and Xiangrong Wang(2020)

[3] Kass, D., Vats, E. (2022) “AttentionHTR: Handwritten Text Recognition Based on Attention Encoder-Decoder Networks”. In: Uchida, S., Barney, E., Eglin, V

[4] "Combining segmentation and recognition for cursive handwriting recognition with multi-dimensional long short-term memory networks" by Puigcerver et al.(2017)

[5] "A hidden Markov model-based handwritten word recognition system for multi-writer Arabic cursive script" by Al-Hajj et al. (2005)

[6] "Cursive Character Recognition with Convolutional Neural Network." by Simistira et al. (2016)

[7] "A hybrid segmentation-free approach for multi-script cursive handwriting recognition with convolutional recurrent neural network" by Ahmed et al. (2020)

[8] "Adapting Tesseract for Cursive Handwriting Recognition through Image Pre-processing" by Arivazhagan et al.(2017)