Malayalam Handwritten Text Recognition using Machine Learning

S Aswathy, V Aswin, K P Muhammed Nishad, M Visakh,
Eighth Semester , 2018 Admission
Department of Computer Science and Engineering,
Sreepathy Institute of Management & Technology, Vavanoor, Palakkad, India-Pin 679533
E-mail: sethumadhavanaswathy@gmail.com, aswinvharidas@gmail.com, nishadshanu07@gmail.com, vishakh10000@gm

***Abstract*— Handwritten text recognition (htr) for Indian languages is not yet a well-studied problem. This is primarily due to the unavailability of large annotated datasets in the associated scripts. Existing datasets are small in size. They also use small lexicons. Such datasets are not sufficient to build robust solutions to htr using modern machine learning techniques.**

Despite the advent of deep learning in computer vision, the general handwriting recognition problem is far from solved. Most existing approaches focus on handwriting datasets that have clearly written text and carefully segmented labels. In this paper, this project instead focus on learning handwritten characters from maintenance logs, a constrained setting where data is very limited and noisy.the problem is broken into two consecutive stages of word segmentation and word recognition respectively, and utilize data augmentation techniques to train both stages. Extensive comparisons with popular baselines for scene-text detection and word recognition show that our system achieves a lower error rate and is more suited to handle noisy and difficult documents.

This method presents region-based, fully convolutional networks for accurate and efficient text detection. In contrast to previous region-based detectors such as Fast/Faster R-CNN (Region with Convolutional Neural Networks) [4] that apply a costly per-region subnetwork hundreds of times, our region-based detector is fully convolutional with almost all computation shared on the entire image. To achieve this goal, we propose position-sensitive score maps to address a dilemma between translation-invariance in image classification and translation-variance in object detection.

Handwriting recognition is a notoriously difficult problem in Machine Learning. Despite decades of research and development, modern handwriting recognition systems still exhibit suboptimal performance in the real world applications. Recent studies show great potential of Recurrent Neural Networks (RNN) [4]with Long Short-Term Memory (LSTM) [16] for unsegmented handwritten word recognition.

I. Introduction

Character recognition by machines is very important in many situations. Many old documents [6] are in the form of hard copies. To make it a soft copy OCR is the method used. But if the case is different like when the document is in handwritten format, it will be very difficult to recognize.

Handwriting recognition can be either online or offline [8]. The former case needs pen trajectory movements and in later case features of the image are considered for transcription. As a subsidiary of online recognition the method of in-air is also getting popularity, where the movement of the finger can translate to a character or a word. Recognition of offline handwriting remains an open research problem for decades. Some extensive studies of the problem, have led to remarkable success in recent past on a number of scripts [5] of developed countries. Also, several handwriting recognition studies of a few Indian scripts such as Devanagari, Bangla, Tamil, Oriya, Malayalam etc. are found in the literature. A survey of handwriting recognition of Indian scripts can be found in the studies of Pal et.al. Although a majority of the works on Indian scripts [6] considered only isolated characters, a few others considered offline handwriting recognition problem in word level. Recognition of offline handwritten Malayalam words has not been explored by the researchers. Also, to the best of our knowledge, there is no available benchmark samples dataset of handwritten Malayalam words for comparison purposes. Handwritten image transcription is an important application of Image Processing, Pattern Recognition and Machine Learning. In Malayalam handwritten texts, the characters are not well separated, but the most of the words are separated by space. So the recognition of handwritten text on the real time data can start with word is an obvious choice.

The character of a human being can be identified through the style of the handwriting because it is unique and represents the individuality of a person. The problem of handwritten recognition is not easy when compared with machine printed text. The major difficulty arised is the variability in the shapes of the characters in the documents, when the same person writes in different situations. In some extreme cases even humans feel difficult to identify and read the documents properly. There exists several scripts and languages across world. The recognition process can be either script dependent or independent. In this work we focus on script independent character and word recognition system with Malayalam as the prime language considered for the recognition.

There are two ways to computerize the old printed malayalam books by means of computers for the facilitate discussion and research:

- Enter those printed books manually and this is very expensive and requires great efforts that the abilities fails to do because there are millions of books that have scientific and historical value.
- Entering the printed books using scanners and then using software for recognizing symbols. This way seems faster and appears more economic.

Here appears the importance of developing that can perform the task of recognition precisely. Most of the methods for pattern recognition, especially template matching techniques, are over sensitive to shifts in position and distortions in shape of the stimulus patterns, and it is necessary to normalize the position and the shape of the stimulus pattern beforehand. Therefore, the finding of an algorithm for pattern recognition which can cope with shifts in position and distortions in shape of the pattern has long been desired. There is a new algorithm that gives an important solution to this problem. This algorithm can be realized with a multilayered network consisting of neuron-like cells.

Malayalam is one among the 22 scheduled languages of India. It is the official language of Kerala and is spoken by around 35 million people in the world. Malayalam is also spoken in the Union territories of Lakshadweep and Mahe [13]. It is one among the 4 major Dravidian languages of South India. Malayalam script is derived from the Grantha script which is an inheritor of the old Brahmi script. Malayalam has close affinity to Tamil. The Malayalam language was written in Vatteluttu, a script that had evolved from Tamil-Brahmi. It first appeared in Vazhappalli inscription. Malayalam has the largest character set among the Indian languages. It is partially alphabetic and partially syllable based. There are about 128 characters. The basic characters can be classified as vowels/Swaraksharangal (Figure 3) and consonants/Vyanjaksharangal. The complete character set includes 15 vowels (Figure 1.1), 36 consonants (Figure 1.2), 5 chillu , 3 consonant signs, 9 vowel signs, anuswaram, visargam, chandrakkala and 57 conjunct consonants. It also includes 9 numerals which are rarely used [13].

Fig. 1. Malayalam Vowels

II. Motivation

The purpose of this project is to take Malayalam handwritten documents as input, recognize the text and modify the handwriting such that it is a beautified version of the input.

Handwriting recognition : Lots of work has been done in the field of character recognition but not much for analyzing a complete document. Recognizing the text of a document would be useful in many diverse applications like reading medical prescriptions, bank cheques and other official documents [5]. It will also find uses in detective or police departments in applications like handwriting based person identification, identifying real from forged documents, etc.

Handwriting recognition can be broken into a number of relatively independent modules. After going through several papers and web pages on handwritten word recognition, we thought of various strategies for each of these modules [11]. We then considered the accuracy and efficiency of these strategies independently and as a whole. Also, we came up with a new idea based on feature extraction and relative position matching with the help of directional graphs. Our project is aimed at implementing this idea keeping in mind the strategies which we considered best for an overall accurate, efficient and scalable handwriting recognition software.

Fig. 2. Malayalam Consonents

The following factors motivated us to select the present problem as our cur- rent research topic. Evaluation of voluminous handwritten assignments and answer scripts are done manually even in this era of information technolog- ical boom. The Availabilty of automatic handwriting recognition systems assist automation of various important services such as postal, courier etc.. Now-a-days the Government of Kerala is encouraging the use of Malayalam as the official language across the state. Thus, a lot of handwritten official documents are getting created everyday and the possible use of an efficient Malayalam handwriting recognizer should lead to better management of official works through fast retrieval of various information. Another factor for motivation is, according to the nature of Malayalam script [5], it is a challenging task that can be solved through machine learning methodologies.

III. OBJECTIVES

Applications of offline handwriting recognition are numerous: reading postal addresses, bank check amounts, and forms. Furthermore, OCR plays an important role for digital libraries, allowing the entry of image textual information into computers by digitization, image restoration, and recognition methods.

- Develop a benchmarking dataset of Handwritten Malayalam words.
- The main objective of this research is to develop recognition methods for handwritten Malayalam words.
- Develop methods for script independent handwritten recognition.

- To identify the best Deep Learning approach for the recognition of handwritten documents.

IV. PROJECT OUTLINE

The rest of the project report is organized as follows. In chapter 2, literature survey done for this project work is given. Chapter 3 presents the formulation of design. Chapter 4 presents the overall design of the proposed system. Chapter 5 describes the dataset used for the analysis. Chapter 6 deals with the experimental results associated with this project and Chapter 7 brings out the conclusion.

V. LITERATURE SURVEY

A. Theoretical Background

Our inputs are rectangular images of varying sizes containing handwritten sentences, often in unaligned lines and with lots of noise and other irrelevant content such as stamps, signatures and other types of random noise. Our goal is to recognize those relevant sentences, and output the corresponding texts for further data analysis purposes [18].

B. Choice of Two-phase Model

As mentioned above, the design is a two-phase approach segment the entire form into words (in the presence of noisy content) while maintaining their original order, and recognize each word individually. There are many reasons for this approach. First, we have very few annotated samples, thus the generalizability of our model is benefited from the inductive bias of the two stage approach. Second, the difficulties of the forms are unusual. Due to unaligned texts, it is impossible to segment forms into lines without affecting the content as in other HWR methods. Furthermore, like scene-text recognition datasets [1], our forms have many types of noise. Third, this approach is interpretable and easier to train and debug. Finally, it becomes easier to perform parallel training of the two stages across limited resources, allowing for better quality control and modularity in design.

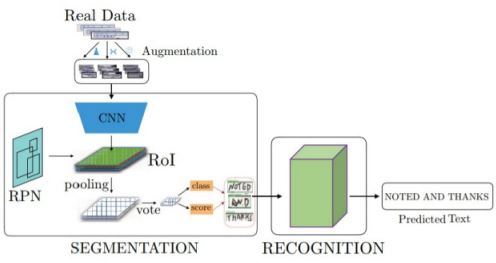

Fig. 3. Architecture

C. Word Segmentation

Instead of trying to predict the correct bounding boxes and recognize the words inside simultaneously, the word segmentation phase only focuses on drawing correct bounding boxes at the word level, and leaves the recognition job as a downstream task. We choose this design for the following reasons. First, word-level segmentation is used since separating spaces among words (as opposed to characters) is much more feasible in practice (especially in cursive handwriting). Second, as explained previously, line-level segmentation is not preferred since in our setting words are often not aligned horizontally.

In terms of architecture, since HWR is different from object detection where detection is only a proxy, we explore multiple options like R-FCN, Faster R-CNN and YOLO-v3 (You Look Only Once : YOLO) to identify which kind of architecture is most suited for our HWR pipeline. Although the core components of those detection methods remain unchanged, it is worth noting two important changes in adapting such methods. First, given word segmentation is an intermediate step, we simplify this phase by limiting the number of classes to only 5, with the main goal being extracting text out of the forms without having to recognize its content. Second, based on the nature of our dataset, we change the segmentation input to grayscale images with only 1 channel. As a result of these two adjustments, our segmentation phase is much easier and faster to train compared with their original uses.

D. Word Recognition

For each form, this module takes the bounding boxes (as images) from the Word Segmentation module as inputs, and outputs a word for each bounding box. Based on the coordinates given by the Word Segmentation module, we are able to reconstruct the entire sentence from individual words [7]. And because the complications of the input forms, we experiment with 3 different models namely Word Model [15], Character Model and CTCSeq2Seq Model, as detailed below.

- Word Model: The word model is a CNN-based [4] image classification network which uses an augmented Resnet-18 to predict words from a predefined word vocabulary. Furthermore, due to the low resolution of our input images, we adjust Resnet-18 to only have a stride size of 1 instead of 2 in the residual blocks. This model is simple, but is only capable of predicting words within the predefined vocabulary of 998 words.
- Character Model: This model shares its architecture with the Word Model, which enables the benefit of initializing weights from a pretrained Word Model, except that it uses a CTC (Connectionist Temporal Classification) loss instead of cross-entropy loss. For this reason, it predicts a sequence of characters instead of a single word at a time. Furthermore, the last fully-connected layer in Resnet-18 is replaced with a convolutional layer to reshape the output from H W D to 1 W/2 C, where C is the cardinality of the character prediction space. By using CTC, this model has two advantages over Word Model. First, CTC largely reduces the prediction space from 998 words to 35 alpha-numeric characters (our dataset does not have the letter “Z”), making it agnostic to word vocabulary size. Second, it enables the model to predict unseen words.

- CTCSeq2Seq Model: Our motivation for this model is to learn the embedded latent representation of images that can be decoded into text. As shown in, the model can be broken down into 3 main blocks: Feature Extraction, Encoder and Decoder. The model loss is the weighted sum of CTC loss (Encoder) and softmax cross-entropy loss (Decoder). Except for those 3 main modules, there is an edit-distance based error module which corrects a predicted out-of-vocabulary word within a maximum of 2 wrong characters compared to a known word.

VI. Existing System

A. Sequence-to-Sequence Contrastive Learning for Text Recognition

Inspired by self-supervised methods for visual representation learning, contrastive learning framework is proposed for sequence-to-sequence visual recognition [15]. To do so,a novel instance-mapping stage that yields a separate instance from every few consecutive frames in the sequential feature map is introduced. These instances then serve as the atomic elements in the contrastive loss. In addition, we design an augmentation procedure that maintains the sequential structure, is crucial for yielding effective representations [19].

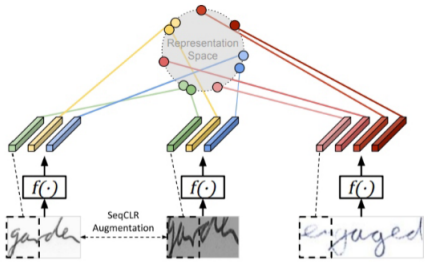

Fig. 4. Sequence-level contrastive(SeqCLR)

As depicted in Fig. 2.2, we suggest a framework consisting of the following five building-blocks:

- A stochastic data augmentation module that is designed to ensure a sequence-level alignment. This operation transforms any given image X_i in a batch of N images, into two augmented images X_i^a , X_i^a belongs to R^{CHWi} , where C denotes the number of input channels, H the image height, and W_i he width of each image which may vary.
- A base encoder f(·) consisting of several blocks of the recognizer scheme. For each pair of augmented images, this component extracts a pair of sequential representations, R_i^a, R_i^b belongs to R^{f*T_i} , where F is the feature dimension, and T_i is the number of frames (columns) which is dependent on the image width.
- An optional projection head g(·), that transforms the representations using a small auxiliary network, as in. We suggest new projection head types that can handle varying sequence sizes, and denote this stage output by P_i^a, P_i^b belongs to $R^{F'T_i}$, where F' is the feature dimension after the projection.
- A novel instance-mapping function m(·) is utilized before the contrastive loss to yield T_i'instances out of T_i projected frames, as illustrated in Fig. These in- stances are then used as the atomic elements in the contrastive loss. Next, we collect all the instances in the batch into two aligned sets Z^a, Z^b , each of size $sum_{i=1}^{N} T_i'$, such that corresponding indices refer to cor- responding frames of the same input image.
- A contrastive loss function as i, that aims to pull closer together representations of correspond- ing indices of z_a, z_b , i.e. positive pairs, and to push all the others, i.e. negative examples, farther apart.
- **Data augmentation**: As pointed out in previous papers, the augmentation pipeline plays a key part in the final quality of the learned visual representations. Current stochastic augmentation schemes are mostly based on aggressive cropping, flipping, color distortions, and blurring. These schemes cannot properly serve the task of text recognition, as they often render the text in the image unreadable. For example, we should refrain from aggressive horizontal cropping as this might cut out complete characters.
  In addition, these augmentation compositions were tailored for tasks as object recognition or classification, where images are atomic input elements in the contrastive loss. However, since in our framework individual instances are part of a sequence, we design an augmentation procedure that ensures sequence-level alignment. Therefore, we avoid transformations such as flipping, aggressive rotations, and substantial horizontal translations. Figure 2.10 depicts differ- ent augmentation types considered in this work, including vertical cropping, blurring, random noise, and different perspective transformations.
- **Base encoder**: The encoder extracts sequential representations from the augmented images X_a, X_b. While in most contrastive learning schemes the identity of the representation layer is pretty clear – usually the visual backbone output, in text recognition there are different options. In particular, we consider two candidates as the sequential representations R_i belongs to R^{F*T_i} , where each option defines f(·) as the text recognizer scheme up to this stage:
  - 1. The visual features,$R_i = V_i$.
  - 2. The contextual feature map, $V_i = H_i$, which better captures contextual information within the sequence.
- Projection head The representations are optionally transformed by a projection head, P_i = g(R_i), which is a small auxiliary neural network that is discarded entirely after the pre-training stage. As indicated in , this mapping improves the quality of the learned representations. Currently, a commonly used projection head is the multilayer perceptron (MLP); however, it can only ac-

commodate fixed-size inputs and thus cannot serve text images. Therefore, we propose two new projection heads in light of the instance-mapping functions defined below: an MLP projection head that operates on each frame independently as the frame-to-instance mapping, and a BiLSTM projection head for improving contextual information in the others mappings

- Instance-mapping Previous work considered images as atomic input elements in the contrastive loss. Therefore, each projected map was vectorized to a single instance, z_i = flatten(p_i). However, the inputs and the feature maps in text recognition are of varying sizes and thus cannot be handled by the flatten operator. More importantly, in text recognition the feature maps have a sequential structure and thus do not represent a single class. Therefore, we propose to view every few consecutive frames in the feature map as an atomic input element for the contrastive loss.
  We propose two approaches for creating individual instances out of sequential feature maps of varying sizes. In the first approach, we transform every fixed number of frames into separate instances, for example, by averaging each W consecutive frames. In the second approach, we fix the number of instances created out of each image, for example, by using adaptive average pooling. In particular, we consider three instance-mapping functions as specifications of these approaches.

B. R-FCN: Region-based Fully Convolutional Networks

1) Two stage detection process:

- Region proposal
- Region classification

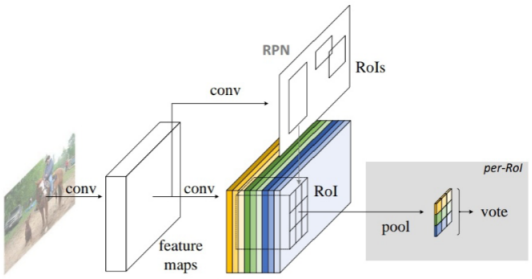

Fig. 5. Overall architecture of R-FCN

Although methods that do not rely on region proposal do exist, region-based systems still possess leading accuracy on several benchmarks [20] . We extract candidate regions by the Region Proposal Network (RPN), which is a fully convolutional architecture in itself. Following, we share the features between RPN and R-FCN. Given the proposal regions (RoIs), the R-FCN architecture is designed to classify the RoIs into object categories and background. In R-FCN, all learnable weight layers are convolutional and are computed on the entire image. The last convolutional layer produces a bank of k.

2) Position Sensitive-Score: Maps for each category, and thus has a $k^2(C+1)$-channel output layer with C object categories (+1 for background). The bank of k^2 score maps correspond to a k × k spatial grid describing relative positions. For example, with k × k = 3 × 3, the 9 score maps encode the cases of top-left, top-center, top-right, ..., bottom-right of an object category. R-FCN ends with a position-sensitive RoI pooling layer. This layer aggregates the outputs of the last convolutional layer and generates scores for each RoI. Unlike, our position-sensitive RoI layer conducts selective pooling, and each of the k × k bin aggregates responses from only one score map out of the bank of k × k score maps. With end-to-end training, this RoI layer shepherds the last convolutional layer to learn specialized position-sensitive score maps.

3) Position-sensitive score maps Position-sensitive RoI pooling.: To explicitly encode position information into each RoI, we divide each RoI rectangle into k × k bins by a regular grid. For an RoI rectangle of a size w ×h, a bin is of a size = w/k × h/. In our method, the last convolutional layer is constructed to produce k^2 score maps for each category. Inside the (i, j)-th bin (0 i, j k 1), we define a position-sensitive RoI pooling operation that pools only over the (i, j)-th score map.

The $k62$ position-sensitive scores then vote on the RoI. In this paper we simply vote by averaging the scores, producing a (C + 1)-dimensional vector for each RoI. They are used for evaluating the cross-entropy loss during training and for ranking the RoIs during inference.

We further address bounding box regression in a similar way. Aside from the above k^2(C +1)-d convolutional layer, we append a sibling $4k^2$-d convolutional layer for bounding box regression. The position-sensitive RoI pooling is performed on this bank of $4k^2$ maps, producing a $4k^2$-d vector foreach RoI. Then it is aggregated into a 4-d vector by average voting. This 4-d vector parameterizes a bounding box as t = (tx, ty, tw, th) following the parameterization in We note that we perform class-agnostic bounding box regression for simplicity, but the class-specific counterpart (i.e., with a $4k^2$C-d output layer) is applicable. The concept of position-sensitive score maps is partially inspired by that develops FCNs for instance-level semantic segmentation. We further introduce the position-sensitive RoI pooling layer that shepherds learning of the score maps for object detection. There is no learnable layer after the RoI layer, enabling nearly cost-free region-wise computation and speeding up both training and inference.

C. Offline Handwriting Recognition Using LSTM Recurrent Neural Networks

1) Preprocessing: The preprocessing phase can be considered as the first stage of the recognition system. The main goal of this step is to modify the images in a way that will make it easier and faster for the recognizer to learn from them [21].

The word images were cut out of high-resolution RGB images. The average width and height of a word image

was 90x60 pixels, with maximum width reaching 300-400 pixels for very long words. Thus 90x60x3=16200 features would be needed to describe an average word image. While images of similar size are nowadays commonly used in image recognition with convolutional networks our main goal was rapid prototyping of various RNN architectures and therefore the size of input data was important factor affecting the total time of training.

Taking into account above mentioned considerations, here applied following preprocessing steps to each word image:

- Dataset augmentation
- Conversion to grayscale
- Resizing

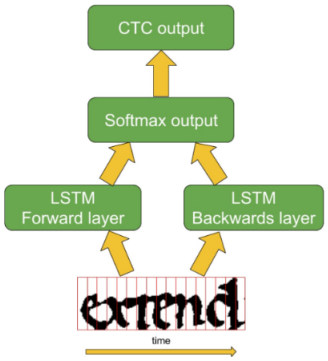

Fig. 6. RNN architecture with Sequence-to-Sequence Learning [20]

Firstly, paper generated 10 additional augmented images by applying random shear (with shear factor from -0.20 to 0.20) to each original image. Secondly, the augmented images were converted to greyscale, on the basis of the assumption that color information provides little discriminative power in the case of handwritten text. Finally, every image was resized to the fixed height (28 pixels) and to the 50 percentage of the original width. This was done to convert the image into format suitable for feeding into RNN where image height would correspond to the size of feature vector and image width to the number of timesteps in the input sequence. This paper considered using convolutional layers for feature extraction, but initial experiments showed that it provided only minor improvement of accuracy and considerably increased the training time. Here also tried simple image enhancement techniques (Gaussian blur for noise reduction and binarization) as additional preprocessing steps, which provided small increase of accuracy for Seq2Seq approach, but did not substantially affect accuracy of CTC approach.

2) Recurrent Neural Networks (RNN): The core part of the approach presented in this work is based on Recurrent Neural Networks (RNN), a particular type of Neural Network that is characterized by the presence of self-connected nodes. These nodes enable the network to memorize and keep track of previous inputs, allowing it to store and access information over long periods of time. This particular ability allows the RNN to learn the context of the labels which turns out to be a big advantage in handwriting recognition since the characters of the words can not be considered as really independent components. In order to use RNNs for handwriting recognition images with handwritten text are interpreted as time sequences along single or multiple axis.

Even though classical RNN provide the network with some memory they are still quite limited in their ability to preserve old content due to the vanishing gradient problem. In fact, as explained by the authors, it is hard for a RNN to bridge gaps of more than about 10 time steps between the moment in which the relevant input is presented and the relative target events. To address this problem we have used Long Short-Term Memory (LSTM). LSTMs are specially constructed RNN nodes to preserve long lasting dependencies. They consist of self-connected memory cell that can be compared to the classical RNN node and three gates that control output and input of the node. Each gate is in fact a sigmoid function of the input to the LSTM node. The first gate is an input gate which controls whether new input is available for the node. The second gate is a forget gate which makes possible for the node to reset activation values of the memory cell. he last gate is an output gate controlling which parts of the cell output are available to the next nodes.

Further improvement to the RNN models based on LSTM is achieved by the use of opposite two- directional layers or so-called Bidirectional Long Short-Term Memory (BLSTM). The goal of the forward layer is to learn context of the input by processing the sequence from the beginning to the end, while the backwards layer performs the opposite operation by processing the sequence from the end to the beginning. It was demonstrated that this architecture performs better than a simple uni-directional LSTM.

D. A Scalable Handwritten Text Recognition System

1) Handwritten Text Line Recognition:

- Model
  The task of handwritten text line recognition is to produce a sequence of Unicode code points for the handwritten text in a single line image [22]. Let x be a line image and y be a sequence of Unicode points. We consider a probabilistic model P(y—x) to represent the relationship between x and y. Here follow the approach described in to model P(y—x): We scale the height of the input image to a fixed size of 40 pixels and collapse the color space down to luminance (gray-scale). The processed image is then fed into a neural network to produce a 1-D sequence of logits where each logit corresponds to a single character or a special blank symbol. The network is trained using a CTC loss. The model naturally handles a variable width (length) of image. At inference time, the logits cost is combined with the cost of a character-based n-gram language model in a log-linear way and beam search is used to find the best recognition result. We experiment with two different model architectures:
  1) LSTM-based model 2) Recurrence-free model
- Training data

The model needs line images (x) along with their transcriptions (y) for training. One way to get such data is to manually annotate handwritten text lines in images collected from various sources. This provides the most valuable data. However, it is time-consuming and costly to do the annotation. Moreover, it is not trivial to find a sufficiently large amount of images containing handwritten text. Thus, the availability of manually labeled data tends to be limited especially when the goal is to support many languages. In the following section describe how we leverage a large amount of stroke d weata that was collected to build an online-handwriting recognition system. This method enables us to support a new script at substantially lower cost.

2) Data Synthesis, Rendering, and Degradation Pipeline: This paper use a mix of different sources of data to obtain the best possible HTR (*Handwritten Text Recognition*) system: We have access to a large amount of ink data collected for building an online handwriting recognition system, in many languages. For the purpose of this paper we are only looking at Latin script but are planning to expand our system to other scripts and languages. We further are using an online handwriting synthesis pipeline to enrich this data with handwriting styles that are not well represented in the original corpus. This data is rendered into images using a rendering pipeline described below and then degraded using the same degrading pipeline that we are using to train an OCR system on synthetic typeset data. To improve recognition accuracy on typeset text, the training data also includes degraded synthetic typeset data. Additionally, we include historic image data from several public datasets to improve recognition accuracy on historical handwriting. Finally, we have a small amount of image data with modern handwriting labeled at the line level.

VII. Formulation of Objectives

A. Problem Definition

Handwriting Recognition is an emerging as well as challenging area in the fields of pattern recognition and computer vision. While optical character recognition (OCR) can be considered as a mature field with very high accuracy rates for printed documents, recognition of handwritten text remains a much harder challenge. Most existing approaches focus on handwriting datasets that have clearly written text and carefully segmented labels. Handwriting recognition technologies has advanced in languages like English, Japanese, Mandarin but Malayalam is still a long way to go. Existing recognition models Malayalam is not very accurate and most requires printed Malayalam texts to work. If a reliable model can be developed for recognizing Malayalam handwritten texts, it can be used to covert many ancient documents, legal documents, and several other manuscripts that are physically stored and are in poor state. Many forms are still being sent out via post where electronic methods are practical. There are a significant number of people who do not have access to a personal computer, the internet, websites and on-line portals and there are some cases where email addresses are not known or don't exist. Applications of offline handwriting recognition are numerous: reading postal addresses, bank check amounts, and forms. Furthermore, OCR plays an important role for digital libraries, allowing the entry of image textual information into computers by digitization, image restoration, and recognition methods.

VIII. Proposed System Outline

Here we are proposing mainly 3 steps for Malayalam character text recognition. Our project uses many algorithms for this purpose. First step: Word segmentation is the process of detecting and extracting individual words from a line of text given as the input. Second step: Character segmentation refers to the process of dividing a word or a line into individual characters. Third step: Character recognition of the resulting characters. Atlast we get the output text in an editable format.

IX. Proposed System

X. Baseline For Text Recognition

Despite the significant demand, there are few efficient methods able to tackle this problem due to the difficulty of designing a holistic solution suitable across various forms of input. The first challenge is to segment the words. The second challenge is to build a model capable of recognizing and generalizing diverse handwriting styles.

A. Segmentation

Segmentation is an operation that isolates individual characters from the handwritten text. It is done using projection profile analysis and connected component labelling. Segmentation includes the following steps:

- Line Segmentation
- Word Segmentation
- Character Segmentation

A popular technique used for line segmentation is horizontal projection profile method in which peak-valley points are identified and is used for line separation. The horizontal projection will have separated peaks and valleys if the lines are well separated. These peaks can be used to find out the boundaries between lines and so each line is extracted. Word segmentation is done by applying vertical projection profile method on the separated lines. Here the peaks and valleys are identified and words are separated by looking at the minima in the vertical profile. Finally, the characters are isolated from these words using connected component labeling. Connected components labeling groups pixels of an image into components based on pixel connectivity. This technique assigns to each connected component of a binary image a distinct label.

*1) **Word Segmentaion**:* Word segmentation is the process of detecting and extracting individual words from a line of text given as the input (Figure 3.1). Word segmentation can be considered an intermediary between line and character segmentation as most of the OCR models are based on character classification. Word classification is not an established field.

Fig. 7. Input for Word Segmentation.

Use vertical projection (VP) profiling to exploit the fact that the distance between two words is greater than the distance between individual characters in a word. A threshold is set and if the distance between two characters is greater than the threshold, it indicates the boundaries of two words. To get word boundaries, VP values are used. Each line is divided into vertical stripes and the number of zeros in each stripe is calculated. If at any point, the number of consecutive zeros is greater than the threshold value, then it is taken as the word boundary. The threshold for word segmentation is also determined using VP method. The counts of consecutive columns with zero VP values are stored, and the mode of all the count values is obtained. The count values greater than 5 times and less than 12 times the mode are then chosen. The average of these count values is then set as the threshold value. The major drawback of this method is the assumption that the spaces between words are greater than 5 times and less than 12 times the spaces between characters. This assumption doesn't always work.(Figure 3.2)

Fig. 8. Result after Word Segmentation.

*2) **Character Segmentation**:* Character segmentation refers to the process of dividing a word or a line into individual characters(Figure 3.3). Unlike line or word segmentation, the same approach can't always be used for character segmentation of several languages because different languages use different types of characters [12]. For instance, Malayalam has special kinds of diacritics like the virama (crescent mark), which are not present in English.

This paper explores a new contour analysis approach in tandem with morphological operations. Rectangular bounding boxes are used to enclose the contours. Unlike projection profiling, contour analysis provides high accuracy even when skewed lines are present. Also, there is no out-of-order segmentation as in the connected component approach.

Fig. 9. Character Segmentation.

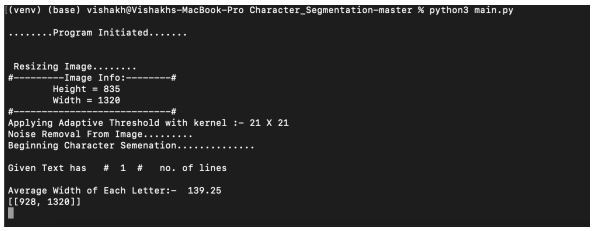

Fig. 10. Image Properties.

B. Using LeNet for Malayalam Character Recognition

Our system of Malayalam character recognition can be divided into two stages as following. We will start with main body recognition using LeNet [7], and after that, dots recognition using search, as the following. Firstly,[]noise will be isolated from the input character. Then, the search area will be determining dependently on the type of shape, some areas will be above the main shape and others below it [11]. And lastly, searching the determined search area in the isolated noise array for single, double, or triple dots depending on the shape of character.

*1) **Network Architecture**:* The network is represented in Figure. Its architecture is a direct extension of the one proposed in. The network has two hidden layers named H1 and H2. H1 is composed of 6 groups of 64 units arranged as 6 independent 8 by 8 feature maps. Each unit in a feature maps takes input on a 5 by 5 neighborhood on the input plane. The motivation is high resolution may be needed to detect the presence of a feature, while its exact position need not be determined with equally high precision. It is known that the kinds of features that are important at one place in the image are likely to be important in other places. Therefore, corresponding connections on each unit in a given feature map are constrained to have the same weights. On other words, each of the unit in H1 uses the same set of 25 weights. The function performed by a feature map can thus be interpreted as a nonlinear sub sampled convolution with a 5 by 5 kernel.

Of course, units in another map share another set of 25 weights. Units do not share their biases. Each unit thus has 25 input plane take their input from a virtual background plane whose state is equal to constant, predetermined background level. Layer H1 comprises 384 units (8 by 8 times 6), 9984 connections (384 times 25) plus 384 biases, but only 534 free parameters (384 biases plus 25 times 6 feature kernels) since many connections share the same weight.

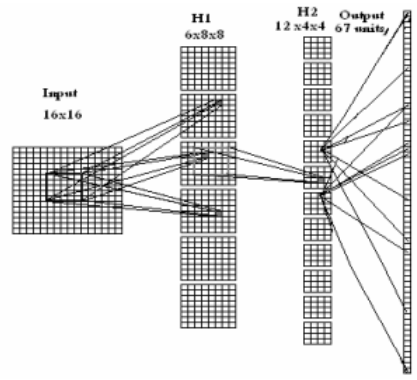

Fig. 11. Connections on the Network.

Layer H2 is composed of 12 features maps [14]. Each feature map contains 16 units arranged in a 4 by 4 plane. The connection scheme between H1 and H2 is quite similar to the one between the input and H1. Each unit in H2 combines local information coming from 4 of the 6 different feature maps in H1 [9]. Its receptive field is composed of six 5 by 5 neighborhoods centered around units that are at identical positions within each of the 6 maps.

Connections falling off the boundaries are treated like as in H1. To summarize, layer H2 contains 192 units (12 times 4 by 4) and there is a total of 19392 connections between layers H1 and H2 (192 units times 101 input lines). All these connections are controlled by only 792 free parameters (6 feature maps [10]. The output layer has 68 units and is also fully connected to H1, it consists of 13124 weights. In summary, the network has 900 units, 42500 connections, and 14450 independent parameters [14].

XI. System Architecture

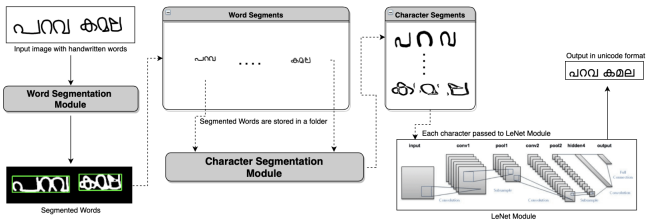

Fig. 12. Architecture.

The system architecture is designed to take inputs of handwritten words as images and the system outputs the words into a text file in the unicode format which can be processed just like regular digital text. The Architecture mainly consists of two phases - the segmentation phase and the recognition phase. The segmentation phase takes the handwritten image as input to the system through the interface, which is then processed by the Words Segmentation Module. The module processes the input image and searches for the words in them and segments them accordingly of their position in the page and stores them in their order of occurrence in a folder. This folder contains all the segmented words. The next step in the segmentation phase is segmenting out each letters from the already segmented words. The Character Segmentation Module takes each word from the Segmented Words folder and segments each words into it's constituent characters and stores it in a folder in a word by word order with appropriate naming so that each character's position can be identified using it's ID. The next phase in the system is the recognition phase. The recognition module is built using LeNet CNN. The original LeNet model was designed primarily to recognize handwritten characters and printed characters, and achieved good results. The model structure is divided into an input layer, a convolution layer, a pooling layer, a fully connected layer, and an output layer, and the input layer is removed, and a total of 7 layers are included. The input is represented by x, usually a matrix or a picture, and the size is 32*32 pixels. The first and second layer convolution layers use a 5*5 size convolution kernel (convolution filter) with a sliding step size of 1, and the third layer convolution layer uses a 1*1 size convolution map. The sliding step size is 1. The pooling layer uses a window of size 2*2 and a step size of 2 for average pooling. The final output uses the softmax algorithm for 10 classifications. The character is then recognised based on the output of the softmax layer. Each of the characters recognised is appended to a text file which eventually gives all the words of the original handwritten input in it's unicode format.

XII. Dataset

The demo dataset was collected from internet and some of the images used for datasets are written by the students of Sreepathy Institute of Management and Technology. Some information about the dataset is shown in the figure below. Before starting handwritten data collection, the Malayalam character classes are decided based on the unique orthographic structures in the Malayalam language script. 85 Malayalam character classes representing vowels, consonants, half-consonants, vowel modifiers, consonant modifiers and conjunct characters that are frequently used while writing is considered for database creation. For collecting character images, the writers are instructed to write the considered Malayalam character classes on pages five times using ballpoint pens by keeping attention on space between each written character. No restriction is kept on the type or quality of the paper and the ballpoint pen used for writing [1].

The handwritten data collected from 77 (60 Female and 17 Male) native Malayalam writers between 20 to 55 age groups and all the writers have minimum graduation as the educational qualification. The learning and testing data are divided based on the writers rather than the collected images. Among 77 writers, the handwritten data collected from 59 persons considered for creating learning (training and validation) data while handwritten data from the remaining 18 persons considered for creating the testing data [17]. Each image is converted to 32*32 dimension.

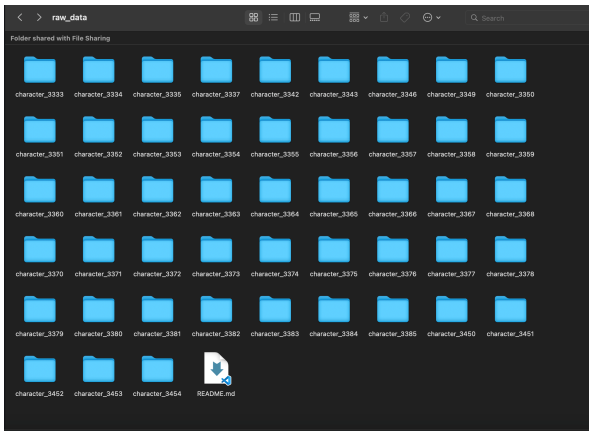

Fig. 13. Datasets.

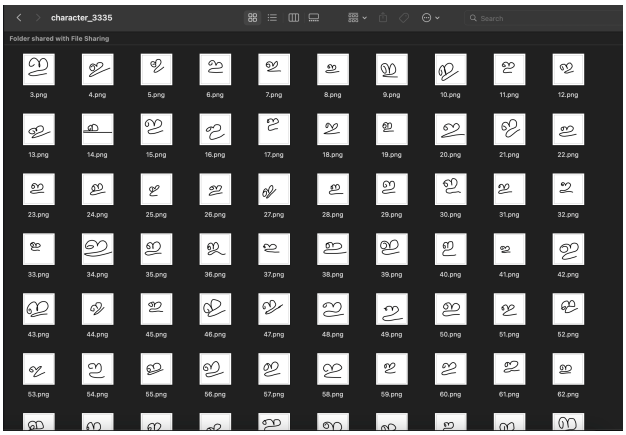

Fig. 15. Character ee dataset.

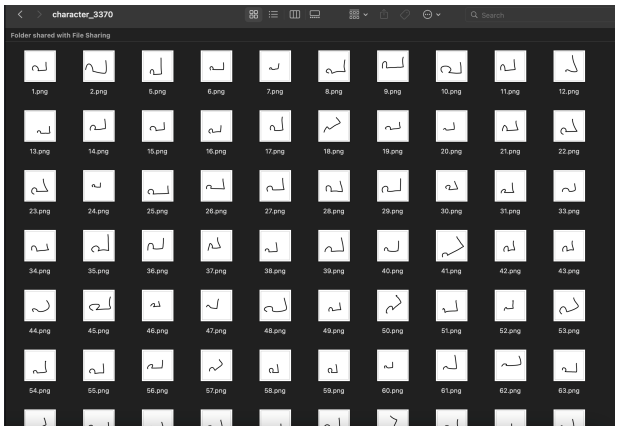

Fig. 14. Character Pa dataset.

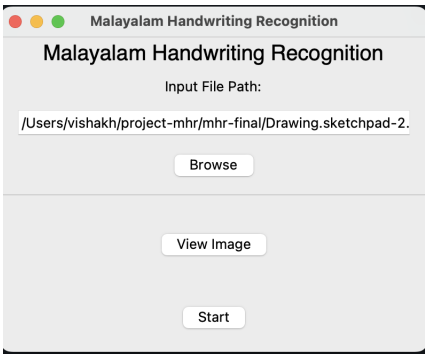

Fig. 16. Interface.

XIII. Experimental Results

A. Performance Analysis

This project is to Recognise Malayalam Handwritten text using Machine Learning. Handwriting recognition, often referred to as optical character recognition, is performed after the writing is completed by converting the handwritten document into digital form. This is the stage in which data are collected as part of the recognition process. The data may be captured online while the user is writing on a digitizer or PDA. In the offline case, the data is obtained by scanning the image after the writing process is over. The Figure 6.1 shows the interface to upload the input document. We can browse it from the files. Figure 6.2 shows the image to upload as the input.

After pressing the ”START” button we get the output as shown in the Figure 6.3. The output will be in a .txt file format and that can be editable.

B. Internal Processes

The aim of preprocessing is to remove as much as distortions as possible from the scanned image. Degraded documents or poor quality of scanner are responsible for these distortions. At first, the scanned image is converted to grayscale if it is in any other format and then the the grayscale image is converted to binary. In order to remove the noise present in the image, a 3x3 median filter is used.

The first process is to segment individual words from the group of words using Bounding boxes as shown in the Figure 6.4.

The word is segmented into individual character. Character Segmentation is the process that isolates individual character from the word shown in the Figure 6.5 and 6.6.

The model is able to predict the above character accurately with 100.0% confidence and it can also predict the whole Malayalam characters with 100% accuracy.

XIV. CONCLUSION

This thesis decribes the new techniques for offline handwriting recognition of Malayalam strings. Word recognition using holistic and analytic approach are discussed. Holistic approach are suitable for limited lexi- con size applications like Town/ Village/ Corporation/ Panchayath name recognition etc. For a generic recognition the approach can be analytic or hybrid. Sequence wise analysis and labelling is a better option for generic recognition and the system is scalable in this case. Several subproblems or challenges for the transcription of handwritten image are addressed in this thesis. The major challenge for the recogntion of Malayalam or any Indian language is to devise a proper method for segmentation or follows segmentation free approach.

Fig. 17. Input Image.

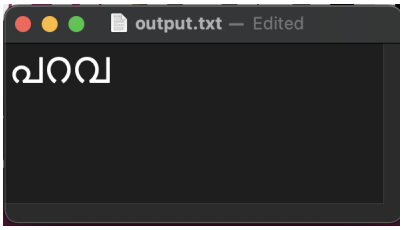

Fig. 18. Output.

Various applications of handwriting recognition are answer paper evaluation, reading notes, bank cheque processing, address interpretation from the postal mails, application form/invoice processing, signature verification, writer identification, gender classification, keyword spotting and medical applications viz. to identify the progress of paralysed patients after the treatment. The model presented a two-stage approach that can process the entire forms directly without the need of segmenting them into lines. The experimental results show that this approach significantly outperforms the HWR and scene-text detection and recognition baselines on the full pipeline while achieving high accuracies on the individual phases of word segmentation and recognition.

Some of the future works are, in practical applications like postal mail, lecturer notes etc., it is common that different scripts are used in the same pages or documents. Script identification and recognition is an obvious solution for it. Another work that can be done along with this is writer identification, it will be useful for various forensic and demographic inves- tigations. Malayalam handwritten keyword spotting is another area to be focussed, which will enhance the abilities to reteieve the right information quickly. Enhanced Label embedding closely related to PHOC(Pyramidal Histogram Of Characters) for Indian languages is also another future area of research. The recognition of multi word, line, Paragraph, page are another area of research to be focus.

REFERENCES

[1] Alaei, A., Pal, U., Nagabhushan, P.: Dataset and Ground Truth for Handwritten Text in Four Different Scripts. IJPRAI (2012).

[2] Baek, J., Kim, G., Lee, J., Park, S., Han, D., Yun, S., Oh, S.J., Lee, H.: What Is Wrong With Scene Text Recognition Model Comparisons? Dataset and Model Analysis. In: ICCV (2019)

[3] Bluche, T., Ney, H., Kermorvant, C.: Feature extraction with convolutional neural networks for handwritten word recognition. In: ICDAR (2013)

[4] Dutta, K., Krishnan, P., Mathew, M., Jawahar, C.V.: Improving CNN-RNN hybrid networks for handwriting recognition. In: ICFHR (2018).

Fig. 19. Word Segmentation.

Fig. 20. Character Segmentation.

[5] Dutta, K., Krishnan, P., Mathew, M., Jawahar, C.: Towards Spotting and Recognition of Handwritten Words in Indic Scripts. In: ICFHR (2018).

[6] Granet, A., Morin, E., Mouch'ere, H., Quiniou, S., Viard-Gaudin, C.: Transfer Learning for Handwriting Recognition on Historical Documents. In: ICPRAM (2018).

[7] Øivind Trier, Anil K. Jain, and Torfinn Taxt. Feature extraction methods for character recognition - a survey. Pattern Recognition, 29:641–662, 1996.

[8] R. Plamondon and S. N. Srihari. On-line and off-line handwriting recognition: A comprehensive survey. IEEE Trans. Pattern Anal. Mach. Intell., 22(1):63–84, 2000.

[9] U. Bhattacharya, S. K. Ghosh, and S. Parui. A two stage recognition scheme for handwritten Tamil characters. In 9th International Conference on Document Analysis and Recognition (ICDAR 2007), volume 1, pages 511–515, 2007.

[10] T. K. Bhowmik, S. K. Parui, U. Bhattacharya, and B. Shaw. An HMM based recognition scheme for handwritten Oriya numerals. In 9th International Conference on Information Technology (ICIT '06), pages 105–110, 2006.

[11] U. Pal and B.B. Chaudhuri. Indian script character recognition: A survey. Pattern Recognition, 37(9):1887 – 1899, 2004.

[12] Segmenting Characters from Malayalam Handwritten Documents. 2019 1st International Conference on Innovations in Information and Communication Technology (ICIICT).

[13] Anitha Mary M.O. Chacko and Dhanya P.M, HANDWRITTEN CHARACTER RECOGNITION IN MALAYALAM SCRIPTS – A REVIEW. International Journal of Artificial Intelligence Applications (IJAIA), Vol. 5, No. 1, January 2014

[14] Neena K Pius1 and Mrs.Alphonsa Johny, Handwriting Arabic Character Recognition LeNet Using Neural Network. The International Arab Journal of Information Technology, Vol. 6, No. 3, July 2009.

[15] JINO P J, Offline Handwritten Malayalam Word Recognition using Machine Learning Techniques. Ph.D Thesis submitted to COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY, 2018.

[16] Dhaval Salvi, Jun Zhou, Jarrell Waggoner, and Song Wang, Handwritten Text Segmentation using Average Longest Path Algorithm.

[17] Santhoshini Gongidi and C V Jawahar, iiit-indic-hw-words: A Dataset for Indic Handwritten Text Recognition. IIIT Hyderabad, India.

[18] Hai Pham†, Amrith Setlur†, Saket Dingliwal†, Tzu-Hsiang Lin†, Barnabas P oczos † Kang Huang, Zhuo Li, Jae Lim, Collin McCormack, Tam Vu. Robust Handwriting Recognition with Limited and Noisy Data. 2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR).

[19] Aviad Aberdam, Ron Litman, Shahar Tsiper, Oron Anschel, Ron Slossberg, Shai Mazor, R. Manmatha and Pietro Perona. Sequence-to-Sequence Contrastive Learning for Text Recognition.

[20] Jifeng Dai, Yi Li, Kaiming He and Jian Sun. R-FCN: Object Detection via Region-based Fully Convolutional Networks. 30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

[21] Matthia Sabatelli and Yaroslav Shkarupa. Offline Handwriting Recognition Using LSTM Recurrent Neural Networks. University of Groningen Department of Artificial Intelligence and Cognitive Engineering Nijenborgh 4, 9747 AG Groningen The Netherlands.

[22] R. Reeve Ingle, Yasuhisa Fujii, Thomas Deselaers, Jonathan Baccash, Ashok C. Popat. A Scalable Handwritten Text Recognition System. 2019 International Conference on Document Analysis and Recognition (ICDAR).

Fig. 21. Individual Character Recognition.

```
##########**##########
Imagefile =  1_1_1.png
Character =  3370
Confidence =  100.0 %
```

Fig. 22. Character Recognition Terminal Outputs.