2022 International Conference for Advancement in Technology (ICONAT)
Goa, India. Jan 21-22, 2022

Offline Text Recognition Using Hidden Markov Model

Sandeep Shiraskar and Sanjeev Patel
Department of Computer Science and Engineering
National Institute of Technology Rourkela,
Odisha - 769008, India
{sandeepshiraskar@gmail.com | patels@nitrkl.ac.in or patelcs01@gmail.com }

***Abstract*—In this paper, we have proposed a methodology of implementing a system model based on Hidden Markov Model (HMM) that can effectively recognize digital textual material. The idea behind the model relies on the ease of implementing HMMs to predict the succeeding character depending on the observable of the present. A similar concept is then applied as a whole for complete word detection and recognition. The model is termed as H and relies on heavily pre-processed images of digital textual data-set. The training phase depends heavily on the vocabulary fed to the system in image format and a series of textual characters, sentences and non-sentimental phrases in text format. Evaluation of the model is expressed in terms of the likelihood of occurrence of testing data. The evaluation result is maintained as the final criterion for the model's ability to filter text from noisy text images. The applications of the project lie in the noise removal from text, clarification of text, scaling the model to operate on a huge amount of textual data and the scope of the project is limitless in image processing and natural language processing.**

***Index Terms*—Baum-Welch Algorithm, Feature selection, Feature extraction, Hidden Markov Model, Viterbi Algorithm, Optical Character Recognition, and Segmentation.**

I. Introduction

The design of a system for text recognition is not a new domain. A massive work has been done and several models have been proposed that are related to universal languages' modelling or region-specific languages wherein scripts have been collected, observed, analysed and recognised for subjective uses. The introduction to this field, defined by a prerequisite list of works, is a vast categorisation. Especially with the evolving mechanisms for generating new models, so is the amount of text that needs to be worked upon increasing. Usually there are three different types of works done in this domain of text processing : text detection, text searching and text recognition. Several dependence models and this research study focuses on the last of the three domains. Among this includes noisy text data, scanned images of text [12], [13] and handwritten text. Noisy textual data is hard to preprocess because the accuracy is highly reduced by the fact that the inter-distance between segments is unevenly recognised by the system. Many techniques for text recognition have been presented in [1]–[5]. The Hidden Markov Model(HMM) combined with Baum-Welch Algorithm attempts to solve this issue. In an arbitrary context, the applications of such a model are innumerable. This research study is composed of a similar work taking references from [13] wherein a scalable system model using HMM is designed. Scalability is enhanced using references from [14] so as to accommodate larger amounts of data. In such cases where the amounts of data to be preprocessed is huge, some of the essential features may be extracted. This immensely improves the performance of the system. HMM based model have been proposed in [7], [8], [14], [16], [18], [19].

Text recognition has been an area of interest in the research domain for many decades. Since the inception of automatic computing and invent of algorithms for machine learning, the ability to detect text, recognise it and correct it has gained a massive importance. Today Automatic Text Recognition is a well-established domain and the software designed for its purpose are publicly available for free either online through products offered by Xiaomi, Google, Adobe, Microsoft etc. or offline through innovative products. The algorithms used in text recognition form the foundation of design of text recognising systems. Naïve Bayes, Hidden Markov Model and Decision Trees were one of the first algorithms to be exploited in character recognition. For the most part these techniques recognised text in the absence of noise. Noise detection and removal is a domain of higher complexity and hence, integration of two or more techniques with character normalization/ slant normalization, improvising and using fuzzy systems in conjunction with artificial neural networks (ANN) helps automatic detection of text in noisy samples [17]. Text recognition techniques based on ANN and deep learning are presented in [9]–[12]. A model for performing the recognition in video sequences has been proposed in [6] using LSTM and CNN.

The contribution of the work is presented as follows. The paper presents a design of a system that can effectively recognise text and also an evaluation model for the same. Various systems have been designed that can perform similar tasks with higher accuracy.

The organization of the paper is as follows. Section 2 describes about the system model. The proposed model and methodology are presented in section 3. Section 4 presents the results and discussions. In the last section, we have given a concluding remarks.

978-1-6654-2577-3/22/$31.00 ©2022 IEEE

II. System Model

The most common approaches used for text recognition are HMM, Artificial Neural Networks, Character Normalization etc. Doubly stochastic approach of the Hidden Markov Model has underlying stochastic processes which are hidden and are tagged by other process. Back Propagation Neural Networks have an important role to play in Optical Character Recognition. This approach directly determines either the system succeeds or fails to recognise the image. Normalization of characters can be part of the process of character recognition. In this paper, we have employed Hidden Markovian Chains to represent characters and also used a classification algorithm called Viterbi Algorithm for word recognition.

A. Text Recognition

Text recognition is an exacting problem in natural language processing and pattern recognition. In Optical Character Recognition the task of recognising characters from digital or handwritten sources is daunting enough after realising the various designs and methods existing. In both academic and commercial applications, scholars have used several algorithms of machine learning to explore character recognition. In fact, the diversity of models proposed have done better than ever in broadening our knowledge of the depth of this challenging domain. Methods such as random forests, K-nearest neighbours, support vector machines, neural networks and, cloud based convolutional neural network have been proposed. Successful results have been achieved using different datasets. There are two categories of datasets to classify:

- On-line text: the model concurrently generates output while the text is being written.
- Off-line text: the model generates output on text that is already completely written.

B. Hidden Markov Model

The Hidden Markov Model is a stochastic model for a Markovian Process. It has hidden processes which are stochastic in nature and can only be realised through a chain of observable sequences which are mapped to related symbols. There are several applications of this model: security surveillance, movement analysis, bit sequence detection and language processing. It is also a widely used technique in speech and text recognition. Its potential lies in modelling sequential dependencies. Both speech processing and text recognition bear many similarities, particularly in cases where both are composed of noisy languages, symbol shapes and ambiguous transitions. It is however noteworthy that, speech signals are time varying signals and characters are sequences of segmented images starting from left to right. In our case, text recognition uses this benefit and is performed based on processing feature vectors using left to right continuous density HMMs. The first order discrete HMM can be defined by the parameters [7] where:

- X = (X_0, X_1, ... X_{N-1}), Set of hidden states (latent states, non-visible states)
- V = V_0, V_1....V_{M-1}, Set of possible observable states (observables, visible states, emitting states)
- N – number of hidden states
- M – number of observable states
- A – (hidden) state transition probability matrix
- B – emission probability matrix
- π - initial (hidden) state probability distribution
- T – length of observation sequence
- O = (O_0, O_1, ..., O_{T-1}) – observation sequence
- Q = (q_0, q_1..., q_{t-1}), $q_t \in$ X – hidden state sequence
- $\gamma = \alpha_i$, where α_i = Prob (q_i at t = T) where T is the final state probability

Given a model H, and observation sequence O = (O_1, O_2, O_{T-1}) which is generated by a state sequence Q = (q_1, q_2, ..., q_{T-1}), of length T, observation's probability is:

$$P(O, Q/\lambda) = \sum_{Q} \pi_{q1} F_{q1}(O_1) \prod_{T} P_{q(T-1)qT} F_{qT}(O_T)...., \tag{1}$$

where, $F_{qj}(O_T)$ is the emitting probability of state T from input state q_j and $P_{qi\ qj}$ is the transition probability from state q_i to q_j. HMMs extract information from samples using statistical algorithms. This is in wide contrast to conventional approaches where samples training is done to verify the proposed hypothesis. Also, with increasing samples, performance of HMMs is directly enhanced.

III. Proposed Model

A. Methodology Used

This section describes the problem formulation and the overall methodology used for character recognition. The overall system method utilises the following phases depicted in Fig. 1. In the first step, character image extraction is performed. In

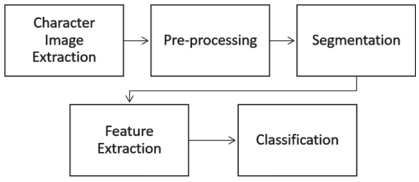

Fig. 1. Flow chart showing the method proposed

this, images of each character of the text are created which are prepared for pre-processing stage. Further, Pre-processing will be carried out in the following sequences. Neighbouring pixels are used to get pixel intensity values in the image. Denoised image is prepared by smoothing the image. New pixel intensity values are assigned by averaging the intensity values of neighbouring pixels. In the next step, segmentation of image will be performed. From the pre-processed image of the character, smallest unit which accommodate a character and which can be just enough for recognition phase, are extracted. Following steps are used in segmentation, (i.) header line location and (ii.) separation of boxed character. In header line location, each row of the matrix is scanned to identify the row

with maximum number of 1s (black pixels). This row is the header line of character. Further, boundary of the character is determined by scanning the matrix column-wise. The column with least 1s is the boundary. The upper (header line) pixels are separated from those in the lower ones. Feature extraction is carried out when image is characterised based on parameters such as width spacing, neighbouring pixels, and segmentation already performed. In classification, each symbol is labelled based on the observable defined by emitting states of the HMM model.

B. Text Recognition System

This sub-section gives a detailed description of the text recognition system which is developed. To begin, a higher-level overview is described, which is followed by in-depth topology of HMM for both characters and words. Since the model is a scaled system, variations in parameters are also fed to the system. Recognition of text requires a systematic approach, which primarily includes determining the correlation between each of the characters. Any character recognition technique can be split into three phases:

1) Pre-processing: Depending on the number of levels the image of the text is represented in a simplified version, i.e. binary levelled image (or black and white image). However, it must be first converted into a grey level image and then into binary image as shown in Fig. 2. For this purpose, pre-processing of images is also called digitization phase. This phase removes most of the noises from the text and hence it is the most important step in the recognition. For example, unnecessary pixels, those having values less than a threshold, are set to level 0. This greatly reduces the improvised image and hence pre-processed templates are then passed on character for segmentation.

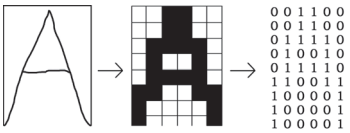

Fig. 2. Digitized image

2) Segmentation: Certain features of the image of the character, such as the position of the object, are noted. This is required so as to assign coordinate pixel values that only represent part of the character that needs recognition.

3) Recognition: Any classification algorithm that goes in conjunction with the employed approach, such as Baum-Welch Algorithm, Viterbi Algorithm, Naïve Bayes etc. are used to correlate the segmented images with templated that have been provided in the dictionary. The dictionary acts as a vocabulary and hence the training set for the system.

The overall topology of the system comprises 2 sections, one following the other. The first one being a classifying function that accepts a complete image of the text as an input and provides characters as outputs. These outputs of characters are supplied as an input to the second classifier that provides words as outputs. Word classifiers used in this project are based on the Forward algorithm which is almost the same as any other character classifier. The second classifier uses the Viterbi Algorithm. Each element in the output set have a classifier containing a Hidden Markov Model. Simply, there will be 26 distinct HMMs for 26 characters. Cross training of all classifiers ensures that they receive inputs from all sets of elements and provide the best possible and accurate output. Fig. 3 shows the classification mechanisms of both classifiers.

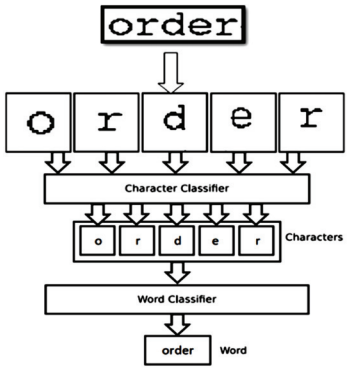

Fig. 3. Representation model of classifiers

If I is an event such that it denotes an input to the classifier, then:

- P(I), is the probability of the event I for all Hidden Markov Models in the classifier:
  - Observation symbols, O_1, $O_2 \ldots$ On are mapped to I. For instance, let I be a string then the classifier outputs a word.
  - For each character, an observation symbol is correspondingly assigned.
  - P(O), i.e. the probability of O is calculated using the forward algorithm.
- Output is the symbol with the highest probability emitted.
- Initial arguments necessitated by the classifier include:
  - Output symbols and training samples
  - A binary variable that is required for training of the system with Baum Welch Algorithm which is also known by Forward-Backward Algorithm.

C. Work Flow of HMM Model

One of the prime assumptions of the Baum Welch Training Algorithm is authenticity of the model. Because of that, prior data is used to decide the topology of the system.

1) HMM Topology for Character Classification: Since there exists a separate HMM for each character given as an output by the classifier, the topology is almost identical to the one suggested by Laan *et al* in [15]. Moreover, the length of the set is limited to only twenty-six characters therefore computation is not much costly. In our experiment, there is a starting and ending state. Therefore, the total number of states becomes $n+2$ if the number of segments produced in feature extraction are n. Since there are multiple observables, the

sequences generated are combined to form a single observation sequence. In Fig. 4, the topology of the sequence is shown where it is provided to Baum-Welch Algorithm as an input. The below listed conditions are fulfilled:

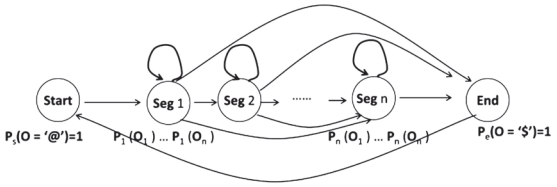

Fig. 4. State transition from Character Classifier

- @ is emitted as a beginning state and $ is emitted as the end state.
- The initial start is transition to the first state
- The final state moves backwards to the first state "start"
- In general, the movement of states is in forward direction by default, considering that the state to be visited has not been visited earlier.

2) HMM Topology for Word Classification: The HMM system for this research study is designed to be scalable in order to accommodate a larger dictionary. Our aim is to design an HMM for each word/character. That shows a tempting step to prove that it is useful when the vocabulary is limited [12]. However, when HMM sequences of each character are concatenated to form a one HMM, a larger vocabulary can be used. In word classification, as discussed earlier, two classifiers are proposed, one based on Forward algorithm and the other one based on Viterbi Algorithm. The latter one is discussed in this section. It provides the flexibility to use only one HMM so that whole dictionary can be accommodated. The state transition for word classifier is presented in Fig. 5. In word classifiers, HMM contains a transition matrix defined

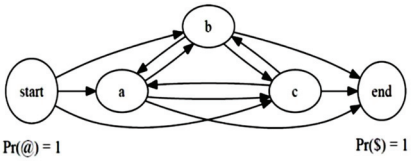

Fig. 5. State transition for word classifier

for all states. For an HMM of 28 states, the transition matrix would be defined as 28 x 28. The following points provide a detailed description of the model:

- Viterbi Algorithm is used to get accurate results of probable states if a string is an input as an observable.
- Similarity index between all possible words and string results are calculated using Hamming Distance
- The word with most similarity is the word that is recognised correctly

3) Initial Parameters: The Baum-Welch algorithm is used to train Hidden Markov Model. It is used in parameter estimation and it is a recursive process. The BW algorithm trains the model and computes expectations to estimate emission and transition probabilities. The process includes both Forward and Backward Algorithm. The BW algorithm is a recursive algorithm and hence the iterations train only the initial model. According to [15] the initialization done were random and random values were assigned in both emission and transition matrices. The information absorbed from training samples was used to assign values to emission matrix in later stages.

4) HMM Evaluation Model: In this research study two methods were realised to evaluate HMM Models:

- Test Sequences Likelihood: Part of the test data was kept aside and likelihood of the test sequences was calculated.
- Prediction of sequences using part of other sequences: The forward algorithm is used to keep record of the HMM states. Computation of P (X_i = n | O_n) where O is an observable, X is a state at i and we are to compute the same X at i+1. Both are adopted as an evaluation mechanism in our proposed system.

IV. Results and Discussions

A. Experimental Setup

1) Pre-processing Feature Extraction: Feature extraction is a process of extraction of observable symbols from training data so that they can be used in image classification. The process of feature extraction is depicted in Fig. 6.

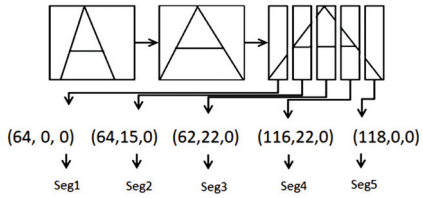

Fig. 6. Representation of feature extraction

The system is built on the assumption that each character has a separated image, a single colour, and inter-linear spacing of at least 1 pixel. The following operations are performed in feature extraction :

- Scaling makes lines thicker and it ensures that the position of the character in the image is independent of the spaces between them. In case where the character occupies a small area of the whole image, the following steps are performed:
  - A small area R shows the position of character in the image.
  - Scaling of R is carried out to fill the vacant spaces. This scaled image is returned as output
- The next step is segmentation of the image into N slices.
- Observable or feature extraction is done by finding the total number of pixels in the segments and by defining a set of colour pixels. We check a connection between neighbouring pixels or connected pixels in a path. Assign-

ing zero values to non-connected pixels or pixels with less than 3 values. Following classification function is used:

$$c_t = \frac{1}{\sum_{i=1}^{N} a_t(i)}, \tag{2}$$

where, a is forward parameter and threshold values of c_t is given as follows:

$$c_t = \begin{cases} \text{Big}, & \text{if } t > d \\ \text{None}, & \text{if } t = 0 \\ \text{Small}, & \text{if } t <= d \end{cases} \tag{3}$$

Hence, classification of all observable states are mapped into either of one from small, big, and none.

2) Dataset: An effort for finding a dataset was done but no existing dataset was found that satisfied the requirements of our work. Instead of choosing a dataset that required a lot of pre-processing, a dataset of text was created. The vocabulary includes 26 uppercase letters, 26 lowercase letters, 9 numerals and 9 special characters. Implementation of slant normalization, skew correction and circulation is required to get the desired output from the dataset.

- Separation of text into small sub images is carried out to equivalent single characters.
- Fixed width and a font are decided.
- Each letter is made for maintaining a certain height-width ratio which is 18 pixels x 24 pixels.
- In the vocabulary, special characters are also included that results to achieve the set to: 26 uppercase and lowercase characters, 10 numbers, space and 7 special characters.
- Baye's Law can be used to estimate observation sequences probability from training data.

B. Testing

The image file containing letters is used as a training set for the program. The program also loads text containing vocabulary. Both of these represent the dictionary for the system. Program uses the classifier to identify the given test file text-image-file.png by using:

- Bayes Rule
- HMM with variable elimination
- HMM with Viterbi

The output is shown in these three forms. The results are achieved using Bayes Net, initial probabilities, size of initial probabilities, tag probabilities, transition probabilities and best matching characters. About 120 test examples are used to test the accuracy of the model. We have performed the results that contain output derived with best accuracy mentioned in terms of probabilities are shown in Table 1.

C. Classification with Varied Parameters

Two arguments are considered for feature extraction in our system model. These arguments are the number of segments that is achieved after the segmentation procedure and Classification Factor. About 100 samples for each 26 letters of alphabet (both uppercase and lowercase) are available. Based on the count and the initialisation before and after the training

TABLE I
TEST FOR CHARACTER CLASSIFIER WITH DIFFERENT INITIALIZATION METHODS AND BEFORE AND AFTER TRAINING

NOE	RIBF	CBIBT	RIAT	CBIAT
90	4%	93%	16%	16%

NOE: Number Of training Examples
RIBT: Random Initialization score Before Training
RIAF: Random Initialization score After Training
CBIBT: Count Based Initialization Score Before Training
CBIAT: Count Based Initialization Score After Training

phase, we use Forward-Backward Algorithm (Baum Welch Algorithm) in our experiment. Based on the initial test count initialisation and random initialisation, we could deduce from the results that the lack of availability of training data had a negative effect on the performance of the system. It may be due to the pre-processing of images and smoothing of images caused loss of certain information which could be used as a model parameter. In this experiment, about 10 percent of the samples were tested against 90 percent of the training samples for each character randomly. The results of this initial test are shown in Table I. We set the values of C and threshold segments as 1.3 and 7, respectively.

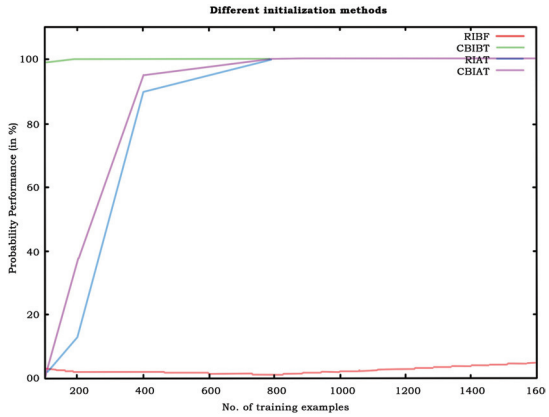

Fig. 7. Count based initialization for different training samples

This experiment measures the performance of the system with varying number of parameters by considering only the initialisation by count. Results presented in Fig. 7 shows that the initial experiment gives best result only when count-based factor is taken into consideration without any training. We have used our own test samples for evaluating the performance of the model. Again, different training samples and test samples are randomly used for each model.

D. Multi-level Classifier with Viterbi Algorithm

We describe the results achieved by using the Viterbi-Classifier in conjunction with the character classifier. We present the results after working out two phased offline text recognition system. The character 'A' is used as a test string and this was provided as an input to the classifier.

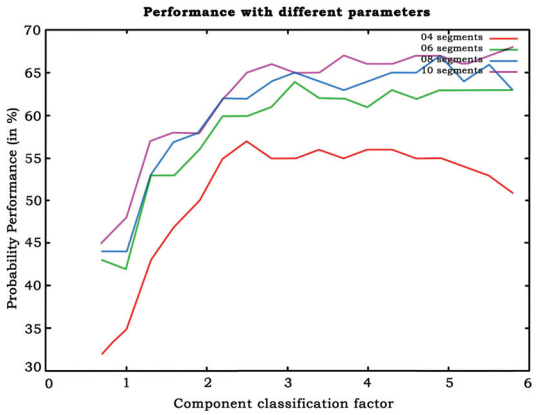

Fig. 8. Performance with different parameters

Different parameters are considered in this sub-section. Fig. 8 presents the prediction probability for different segments. In this experiment, the differences between with and without Viterbi Algorithm for corrections are observed. For each character in the sample, about 10 segments (or 10 samples) are used as test cases. With Viterbi Correction, we achieve the score more than 90%. Without Viterbi, we achieve only 70 percent score against varied component classification factors as shown in Fig. 8.

V. Conclusion and Future Work

In this paper, we have performed the experiment that includes the implementation of a hybrid HMM model with a Viterbi Classifier. The system passes through following phases: Pre-processing, segmentation, character classification, Word classification and Recognition. The proposed model is effectively tested against string samples where the threshold segments are set for the image of each character between 7 to 20, depending on each character. The character classification involved the use of character classification component (C) and segments are mapped to each character. Viterbi Classifier is used for mapping the character to word classification. It also improves the results for character classification. The results show the impact of the size of training on performance of the system. This is a significant observation as limited training data may alter the performance of the system to degrade. The accuracy of the system worsens when Forward Classifier (Baum-Welch Algorithm) is used on a training sample size of less than 800. The Viterbi Classification results shown in Fig. 8 shows that it is very successful in classification of words.

It has been shown from the experiments carried out in this paper that HMM model can be viably used as an alternative for an offline text recognition system. However, the results indicate that a lot of improvements could be made to increase the performance of the system. One proposed direction of future research could be to explore the segmentation and feature extraction procedure. Potentially, feature extraction from both left-right and top-bottom can be performed.

References

[1] A. Graves, "Offline Arabic Handwriting Recognition with Multidimensional Recurrent Neural Networks," In: Märgner V., El Abed H. (eds) Guide to OCR for Arabic Scripts. Springer, London, 2012.

[2] A. Jayakumar, G. S. Babu, R. Raman, and P. Nedungadi, "Integrating Writing Direction and Handwriting Letter Recognition in Touch-Enabled Devices," in *Proc.* ICCCT, Sept. 2015, pp. 393-400.

[3] B. S. Kim, H. I. Koo, and N. I. Cho, "Document de-wrapping via text-line based optimization," *Pattern Recognition*, vol. 48, no. 11, pp. 3600–3614, Nov. 2015.

[4] S. Espana-Boquera, M. J. Castro-Bleda, J. Gorbe-Moya, and F. Zamora-Martinez, "Improving offline handwritten text recognition with hybrid HMM/ANN models," *IEEE Trans Pattern Anal Mach Intell*, vol. 33, no. 4, pp. 767–779, 2011.

[5] O. Morillot, L. Likforman-Sulem, and E. Grosicki, "New baseline correction algorithm for text-line recognition with bidirectional recurrent neural networks," *J Electronic Imaging*, vol. 22, pp. 023028, 2013.

[6] A. Ullah, J. Ahmad, K. Muhammad, M. Sajjad, and S. W. Baik, "Action recognition in video sequences using deep bi-directional lstm with CNN features," *IEEE Access*, vol. 6, pp. 1155–1166, 2018.

[7] P. P Roy, A. K. Bhunia, A. Das, P. Dey, and U. Pal, "HMM-based Indic handwritten word recognition using zone segmentation," *Pattern Recognition*, vol. 60, pp. 1057–1075, Dec. 2016.

[8] A. Ahmad, L. Rothacker, G. A. Fink, and S. A. Mahmoud, "Novel sub-character hmm models for Arabic text recognition," in *Proc.* ICDAR, Oct. 2013, pp 658–662.

[9] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition," *IEEE Trans Pattern Anal Mach Intell*, vol. 39, no. 11, pp. 2298–2304, 2017.

[10] R. Maalej, N. Tagougui, and M. Kherallah, "Online Arabic handwriting recognition with dropout applied in deep recurrent neural networks," in *Proc.* IAPR Workshop on Document Analysis Systems (DAS), Apr. 2016, pp 417–421.

[11] A. Graves and J. Schmidhuber, "Offline handwriting recognition with multidimensional recurrent neural networks," in *Proc.* Advances in Neural Information Processing Systems(NIPS), 2099, pp 545–552.

[12] H. E. Bahi and A. Zatni, "Text recognition in document images obtained by a smartphone based on deep convolutional and recurrent neural network," *Multimedia Tools and Applications*, vol. 78, pp. 26453–26481, Jun. 2019.

[13] I. Z. Yalniz and R. Manmatha, "Dependence Models for Searching Text in Document Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 1, pp. 49-63, Jan. 2019.

[14] J. T. Chiu and A. M. Rush, "Scaling Hidden Markov Languages Models," in *Proc.* 2020 Conference on Empirical Methods in Natural Language Processing, Nov. 2020, pp. 1341-1349.

[15] N. C. Laan, D. F. Pace, and H. Shatkay, "Initial model selection for the Baum-Welch algorithm as applied to HMMs of DNA sequences," in *Proc.* CSCBCE, 2006.

[16] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," in *Proc.* of the IEEE, vol. 77, no. 2, pp. 257-286, Feb. 1989.

[17] R. Plamondon and S. N. Srihari, "On-line and off- line handwritten character recognition: A comprehensive survey," *IEEE. Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 63-84, 2000.

[18] P. Khurana, and V. Singh, "A model for human cognition," *International Journal of Computing and Business Research*, vol. 2, no. 3, 2011.

[19] S. N. Sivanandam and S.N Deepa, "Principles of Soft Computing," *Wiley-India publisher*, 2nd edition, 2011.