ISSN:0975-9646

Handwritten English Character Recognition using HMM, Baum-Welch and Genetic Algorithm

Atmaprakash Singh[1]
Department of Computer Science and Engineering Maharishi University of Information and Technology, Luck now

Ravindra Nath[2]
Department of Computer Science and Engineering, University Institute of Engineering and Technology , Chattrapati Shahuji Maharaj University, Kanpur

Santosh Kumar[3]
Department of Computer Science and Engineering Maharishi University of Information and Technology, Luck now

Abstract-It is the problem of computer science that how we detect the handwritten character and word, so it is also the problem in the field of image processing and pattern recognition of computer science. The meaning of handwritten character and word recognition refers to the identification of the characters or word which is written by a human being. Our approach is this, how this problem solved correctly. In Handwritten character recognition, we have to assign each character into its (A-Z, a-z or 0-9). In this paper we use two approaches Hidden Markov Model (HMM) and Genetic Algorithm (GA) to identify features of each character and compare with its testing set of characters. In this paper we uses various stages of handwritten character recognition system that are: read a scanned image of hand written character, converting this matrix into binary form (0 and 1), resizing each character matrix into size of (n x m where n and m may be same), and thinning of an image to get a clear skeleton of each character. Then In this paper identify the each character using three algorithms namely: Forward Algorithm, Baum Welch and Genetic Algorithm. The results obtained from each of the algorithm are compared separately and at the end the accuracy of these algorithms are compared separately.

Keyword: HMM (Hidden Markov Model), GA (Genetic Algorithm), Baum Welch Method (BWM), Handwritten Character Recognition (HC).

1. Introduction

1.1 Handwritten character recognition:

Character and word recognition refers to the identification of hand written words and characters (machine printed or Handwritten char). The handwritten characters are not always of the same size, thickness, or orientation. Our goal is to implement a model which can classify those characters into appropriate classes. These characters match to appropriate characters of the given data. Offline characters and word recognition refer to the recognition technique in which the final figure is given to us and we have no idea of how the writer wrote the word composed of various characters. Therefore, we implement a model, through which we can identify the written character and word. We can think of character recognition as machine simulation of human writing. First we started with the recognition of isolated handwritten character, in- which we obtain important information for character recognition from the extracted features. We have chosen the HMM model and GA for training because the advantage with HMM based systems is that they are mathematically strong model for the training and testing an image of the character. The task of handwritten character recognition, using a classifier, has great importance and use such as the applications of online handwriting recognition; recognize characters on scanned images of bank related documents. We have faced many challenges while attempting to solve this problem.

1.2 Dataset Used

Some of the handwritten characters are shown below:

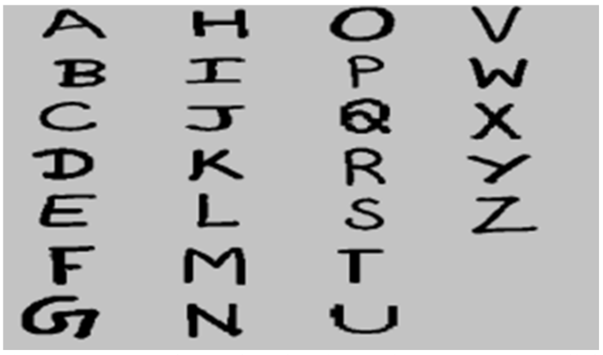

Figure 1.1 Samples of hand written characters of English language

Our goal is the feature extraction of a character to classify the patterns into categories. A well-known example in this field is the handwritten character recognition where characters have to be assigned into one of the 26 classes using some classification method based on recognition techniques. Based on the recognition of characters of English language, we have shown a sample of data that are used as an input.

2. Hidden Markov Model

A Hidden Markov Model (HMM) is a stochastic model that uses the statistical properties of observed real world data. A good HMM accurately models the real world source of the observed data and has the ability to simulate the source. Machine Learning techniques based on HMMs have been successfully useful to applications including speech recognition, optical character recognition, and as we will examine problems in computational biology [1].

Hidden Markov model used for applications to more complex process, including speech recognition and computational gene finding. A generalized Hidden Markov Model (HMM) contains of a finite set of states, an alphabet of output symbols, a set of state transition probabilities and a set of emission probabilities. The emission probabilities require the distribution of output symbols that may be released from each state. [1]

2.1 Mathematical Description of HMM

In order to characterize an HMM completely, following elements are needed [2]

a. The number of states N in the model.

b. The number of distinct observation symbols M per state.

c. The state transition probability distribution A
A = { a_{ij} } where $a_{ij} = p(q_t = S_j | q_{t-1} = S_i)$

d. The observation symbol probability distribution in state j

$$b_j(k) = P(V_k at\ t \mid q_t = S_j)$$

e. The initial state distribution probability

$$\pi_i = P(q_1 = S_i)$$

$\lambda = (A, B, \pi)$.

The Three main problems of HMM are:

Evaluation problem: Decoding problem, and Learning problem. Evaluation problem: Compute P (O | λ), the probability of the observation sequence O = O_1 O_2 O_3…O_T, given the model λ = (A, B, π).

Decoding problem: In this, we attempt to uncover the hidden part of the model i.e. find the optimal state sequence, for the given observation sequence O = O_1 O_2 O_3…O_T, given the model λ = (A, B, π). **Learning problem:** model parameters (A, B, π) are adjusted such that P (O| λ) is maximized.

In this paper, we have evaluated value of P (O| λ) using forward method and training is done using third problem of HMM called learning problem of HMM.

In this paper we are using the third problem of HMM ie. Learning problem of HMM. So we evaluate the values the optimum value of P (O| λ).

3. Preprocessing And Feature Extraction

The pre-processing is a series of operations performed on the scanned input image. It essentially enhances the image rendering it suitable for segmentation. Binarization process converts a gray scale image into a binary image using global threshold technique.

3.1 Preprocessing

It usually consists of binarization, normalization and sampling, smoothing and de-noising of a character image. Any image processing application suffers from noise like isolated pixels etc. This noise gives rise to ambiguous features which results in poor recognition rate or accuracy. Hence we take the data set which is noiseless. After this the image of a character is binarized. Binarization process converts a gray scale image into a binary image. Thinning is performed to get the skeleton of character image so that strokes could be conspicuous. In order to explain the preprocessing phase, we have taken a sample image hand written character **'A'** and then we have applied binarization, resizing and thinning to this scanned image. This process is shown in the figure given below:

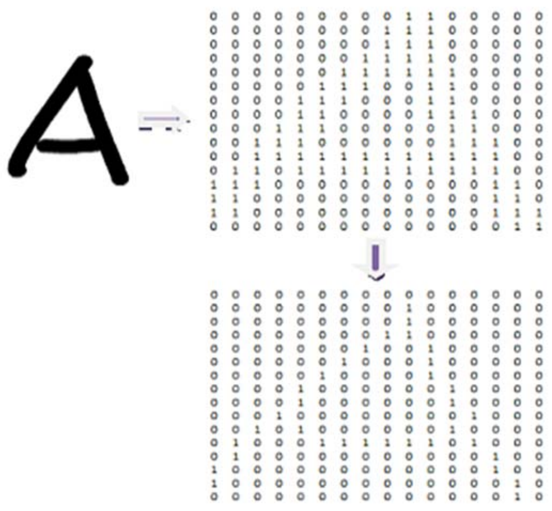

Figure 3.1 Description of a character in the form of pixel and binary digit

Therefore we have obtained a thinned image in binaries form of 16x16 matrices. This image will be processed in the next phase. We see in the above figure that A is written shown by the binary 1 and rest part is 0.

3.2 Feature Extraction

Feature extraction is an important part of any type of pattern recognition. A better feature extraction method may yield better recognition rate by a given classifier. Therefore, much attention is paid to extract the suitable features from the preprocessed images. Our feature extraction process consists of LOCAL FEATURE EXTRACTION METHODS**.** Feature extraction can be defined as the process of extracting distinctive information from the matrices of character images. The feature extraction phase is just as important as the classification phase. In this research work the feature extracted will be used solely as the input for the classification phase. This component is regarded as very important because the focal point of the research lies in the success of the features extracted.

3.3 Process of Feature Extraction

After the preprocessing step, we have obtained a thinned image of given input character. Now we extract the local features from this preprocessed image. The image is divided into eight equal blocks of 4x8, from each block eight gradient features are extracted, each block represents a state in our model, thus we have eight states and the final observation sequence contains 64 observations. Feature extraction phase is the basis of our recognition model. The possibility of matching a character to a particular class depends on how these features are extracted. More accurately the features extracted, more will be the result of recognition.

This process is shown in the following figure:

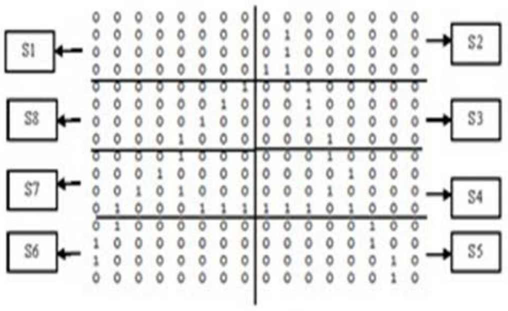

Figure 3.2 Feature Extraction

4. Character Recognition Through Of Hmm

After the feature extraction process, we construct a HMM model for character. We have taken the same image of character 'A' as an example. Before calculating A, B and π, we create two observation sequences. Both sequences consist of 64 observations. As an instance, first sequence is generated as in the form of 1's and 0's.(If number of 1's are more than number of 0's in a column, then value is 1 otherwise 0). These observation sequences are shown below:

O={000000000100000000010000000011000000001101000
00000000100000000000}

Π is taken as [0.4, 0.4, 0, 0, 0, 0.2, 0, 0]

A is the probability of transition from one to other. Initial values are shown in following table:

$A= \{a_{ij}\}$

	S1	S2	S3	S4	S5	S6	S7	S8
S1	0.1	0.2	0.2	0	0	0	0	0.5
S2	0.2	0.1	0.5	0	0	0	0	0.2
S3	0	0	0.1	0.4	0	0	0.4	0.1
S4	0	0	0	0.1	0.5	0.4	0	0
S5	0	0	0	0.6	0.1	0	0.3	0
S6	0	0	0	0.3	0.3	0.1	0.6	0
S7	0	0	0.15	0.15	0	0.3	0.1	0.3
S8	0.3	0.1	0.1	0.1	0	0	0.3	0.1

Table 4.1: Initial Sate Transition Probability Matrix for character A

$B = \{b_j(k)\}$

	Prob of occurrence of 0	Prob of occurrence of 1
S1	1	0
S2	0.875	0.125
S3	0.875	0.125
S4	0.78125	0.21725
S5	0.875	0.125
S6	0.90625	0.09375
S7	0.75	0.25
S8	0.875	0.125

Table 4.2: Observation Probability Matrix for Character A

The value of matrix B changes for each image as it counts the number of ones in the corresponding state during the time of feature extraction. We have used Forward algorithm to train the HMM using observation sequence obtained from the feature vectors. At the end of training process we obtain the final value of A and B which is used for recognition purpose.

4.1 Forward Algorithm

The forward algorithm, in the context of a hidden Markov model, is used to calculate a the probability of a state at a certain time, given the history of evidence. The process is also known as filtering. We want to find the probability of an observed sequence given an HMM - that is, the parameters (π, A, B) are known. Given this algorithm, it is straightforward to determine which of a number of HMMs best describes a given observation sequence - the forward algorithm is evaluated for each, and that giving the highest probability selected.

4.2 HMM Training By Forward Algorithm:

The forward algorithm is implemented for calculating the probability of the various observation sequence mentioned above. Thus using our HMM model parameters, and a sequence of observations, we computed P(O|λ), the probability of the observation sequence given a model. This problem could be viewed as one of evaluating how well a model predicts a given observation sequence, and thus allow us to choose the most appropriate model from a set. We define the forward probability variable:

$\alpha_t(i) = P(o_1 o_2 \cdot\cdot\cdot o_t, q_t = s_i|\lambda)$

So if we work through the trellis filling in the values of α, the sum of the final column of the trellis will equal the probability of the observation sequence. The algorithm for this processing called the forward algorithm and is as follows:

1. Initialization:

$\alpha_1(i) = \pi_i b_i(o_1),\ 1 \le i \le N$

2. Induction:

$\alpha_{t+1}(j) = [\sum_{i=1}^{N} \alpha t(i) a_{ij}] b_{jo(t+1)}\ ,\ 1 \le t \le T-1,\ 1 \le j \le N$

3. Termination:

$P(O|\lambda) = \sum_{i=1}^{N} \alpha T(i)$

For each state s_j , $\alpha_j(t)$ stores the probability of arriving in that state having observed the observation sequence up until time t.

4.3 Training

First Training: For first observation sequence $O=(O_1\ O_2\ O_3\ ....\ O_T)$ we calculate the value of P(O|λ) by using the Forward algorithm each time by altering the A and B matrices simultaneously (We do changes by adding and subtracting some number d in range of (0.001-0.009) from the same row). Thus we generate approximately one thousand values of P(O|λ) and arrange them in descending order and take the best value of P(O|λ). Distinct values of P(O|λ) (in decreasing order) obtained after first training of English character **"A"**

No of Iteration	Values of P(O|λ)
1st	-51.022585926816483
10th	-51.035242759522404
100th	-51.060592622317934
200th	-51.108422095927928
500th	-51.117989499233317
700th	-51.118516686950485
1000th	-51.121374238290144

Table 4.3.1: P(O|λ) values after first training

Best value of the P (O|λ) after first training is **-51.022585926816483** with the corresponding values of matrices A and B as

$A= \{a_{ij}\}$

	S1	S2	S3	S4	S5	S6	S7	S8
S1	0.0970	0.2010	0.2010	0	0.0010	0	0	0.5000
S2	0.1950	0.1000	0.5000	0	0.0010	0.0020	0.0010	0.2010
S3	0	0.0020	0.0960	0.4010	0	0	0.4010	0.1000
S4	0.0010	0.0010	0	0.0950	0.5000	0.4020	0.0010	0
S5	0.0010	0.0020	0	0.5940	0.1000	0.0010	0.3010	0.0010
S6	0	0.0010	0.0010	0.2920	0.0020	0.1010	0.6030	0
S7	0.0010	0	0.1450	0.1500	0.0010	0.3020	0.1000	0.3010
S8	0.2990	0.1000	0.1000	0.1000	0.0010	0	0.3000	0.1000

Table 4.3.2 corresponding A matrix after first training

$B= \{b_j(k)\}$

	Probability of occurrence of 0	Probability of occurrence of 1
S1	0.9940	0.0060
S2	0.8680	0.1320
S3	0.8820	0.1180
S4	0.7993	0.2007
S5	0.8690	0.1310
S6	0.8982	0.1018
S7	0.7430	0.2570
S8	0.8720	0.1280

Table 4.3.3 Corresponding B matrix after first training

4.4 Retraining after using best values of P(O|λ)

For the better result we re-train the data best valu of P(O|λ). We again generate 1000 values of P(O|λ) by Forward algorithm and arrange these values in descending order and take the best value. This is our final P(O|λ) value for that character image. Using these above values of A and B matrices we have re-train and Values of P(O|λ) obtained after second training.

No of Iteration	Values of P(O|λ)
1st	-50.916874337730860
10th	-50.930509802451851
100th	-50.970300429956815
200th	-51.009802352479440
500th	-51.020466996455418
700th	-51.023716664097158
1000th	-51.024407208031974

Table 4.4.1 P(O|λ) values after second training

maximum value of the P(O|λ) after second training is **-50.916874337730860** with the corresponding values of matrices A and B as

$A= \{a_{ij}\}$

	S1	S2	S3	S4	S5	S6	S7	S8
S1	0.0930	0.2010	0.2030	0.0010	0.0010	0	0.0010	0.5000
S2	0.1900	0.1010	0.5010	0	0.0020	0.0030	0.0020	0.2010
S3	0.0010	0.0020	0.0870	0.4020	0.0020	0.0020	0.4040	0.1000
S4	0.0010	0.0010	0.0020	0.0920	0.5000	0.4030	0.0010	0
S5	0.0010	0.0020	0.0010	0.5860	0.1030	0.0030	0.3030	0.0010
S6	0.0010	0.0010	0.0010	0.2880	0.0040	0.1020	0.6030	0
S7	0.0010	0	0.1400	0.1510	0.0020	0.3030	0.1020	0.3010
S8	0.2970	0.1010	0.1010	0.1000	0.0010	0	0.3000	0.1000

Table 2.7 corresponding values of A matrix after retraining

Table 4.4.2 corresponding values of A matrix after retraining

$B= \{b_j(k)\}$

	Probability of occurrence of 0	Probability of occurrence of 1
S1	0.9150	0.0850
S2	0.8660	0.1340
S3	0.8890	0.1110
S4	0.8113	0.1887
S5	0.8600	0.1400
S6	0.8902	0.1098
S7	0.7340	0.2660
S8	0.8720	0.1280

Table 4.4.3 Corresponding B matrix after second training

Similarly we have done above procedure for other nine images of handwritten English alphabet character "**A**" and we get the following are the ten training values of P(O|λ).

Image	P(O|λ)	Difference from best value
1	-50.916874337730860	7.782441889367632
2	-48.279559193022308	5.145126744659081
3	-57.129074100718242	13.994641652355014
4	-51.303209852612092	8.168777404248864
5	-48.474675436437778	5.340242988074550
6	-49.946438538706289	6.812006090343061
7	-43.134432448363228	0
8	-51.038417901906406	7.903985453543179
9	-51.276859060602597	8.142426612239369
10	-51.158288761744011	8.023856313380783

Table 4.4.4 P(O|λ) values for ten images of character A by using forward algorithm

4.5 Testing using forward algorithms

We have tested the six other English alphabet character A images and we get the following values of P(O|λ) after subtracting from the MAX value for character *A* stored at the time of training

For English character "***A***" we have trained our hidden Markova model using the Forward algorithm and get the following range 0 to 13.99464.

Image No.	Final Values of *P(O|λ)*	Value of *P(O|λ)* after subtracting from max value	Char Recognized
1	-43.147111418736593	0.012678970373365	Yes
2	-48.679357477299895	5.544925028936667	Yes
3	-50.467939257568304	7.333506809205076	Yes
4	-47.881750378711182	4.747317930347954	Yes
5	-53.678154950572903	10.543722502209675	Yes
6	-51.996098935198376	8.861666486835148	Yes

Table 4.5: results obtained by forward algorithms

4.6 Analysis using forward method

It is concluded that our forward algorithm gives 99.90% result as it recognizes all the samples of character "**A**" taken above as an example. This algorithm is highly efficient because it provides a strong mathematical ground.

5. Character Recognition Using Baum-Welch Algorithm

The most difficult problem is to determine a method to adjust the model parameters (A, B, π) to maximize the probability of the observation sequence O = [O_1 O_2 O_3,…….,O_T] for the given model. However we can choose λ= (A, B, π) such that P(O|λ) is locally maximized using an iterative procedure such as Baum-Welch method.

5.1 Training through Baum-Welch Algorithm

We have directly used the Baum-Welch method defined in MATLAB for the cross-verification of the Hidden Markov Model that we have created. Functions used

[seq, states]= hmmgenerate (64, A, B);

This function generates the sequence of observation and their corresponding states by using the transition matrix A and symbol emission matrix B

[estTR1, estE1]=hmmtrain (seq, A, B, ’Algorithm’, ’BaumWelch’, ’MAXITERATIONS’, 1000);

This function uses the above generated sequence with the transition matrix A and symbol emission matrix B. It also needs the algorithm to train the Hidden Markov model- we have used the Baum-Welch here, we confined the maximum iteration to 1000 so that the value of P(O|λ) converges easily.

for training we have used the above described procedure until the optimum value of P(O|λ) is calculated. Table listed below shows the optimum values of P(O| λ) obtained after applying the above described procedure are on ten images of English character ***A***;

Image	P(O|λ)	Difference from best value
1	-21.75000548	8.380801449999998
2	-15.04964554	1.680441510000000
3	-16.76982125	3.400617219999999
4	-22.38441968	9.015215650000000
5	-21.47937189	8.110167859999999
6	-15.64546880	2.276264769999999
7	-13.36920403	0
8	-15.28621599	1.917011960000000
9	-17.94169248	4.572488450000000
10	-28.45997322	15.090769189999998

Table 5.1 P(O|λ) values for ten images of character A by Baum- Welch algorithm

English character *A* we have trained our model by using the Baum-Welch Algorithm by utilizing the built-in function hmmtrain (…) of MATLAB and get the following range 0 to 15.090769**.**

5.2 Testing using Baum-Welch Algorithm:

We have tested the six other English alphabet character *A* images and we get the following values of *P*(*O*|λ) after subtracting from the MAX value for character *A* stored at the time of training

Image No.	Final Values of *P*(*O*|λ)	Value of *P*(*O*|λ) after subtracting from max value	Recognized
1	-6.950796732689399	-6.418407297310601	No
2	-14.967798803448392	1.598594773448392	Yes
3	-19.898750944869093	6.529546914869092	Yes
4	-7.792943806460137	-5.576260223539864	No
5	-22.895619938543170	9.526415908543170	Yes

Table 5.2: results obtained by Baum-Welch algorithms

5.3 Analysis through Baum-Welch Algorithm

Above result shows that four values out of six values lie in the range. This shows that our HMM model have the efficiency of 66.67% (approximately) for recognizing the English character *A*.

6. Character Recognition Using Genetic Algorithms

6.1 Genetic Algorithm

Genetic Algorithm is a search technique that imitates the natural selection and biological evolutionary process. GA has been used in wide variety of applications, particularly in optimization problems. In a genetic algorithm, we start with a set of feasible solutions, called population, then the crossover and mutation operations are applied to pair of solutions and this way a new population is generated. We continue this process to get a final solution. Genetic algorithm is a randomized search algorithm and it can be outlined as follows. [29]

6.2 Training through Genetic algorithms

In Implementation of Genetic Algorithm, we have taken the state transition matrix A used in Hidden Markov Model (as described earlier). The *crossover* operation is applied on the state transition matrix A to generate the random population. The value of P(O|λ) are calculated for all these random populations. The state transition matrix A corresponding to the maximum value of P(O|λ) is further taken for creating the next set of population by using crossover and mutation operation simultaneously. After the creation of these populations again value of P(O|λ) are calculated and the corresponding value of state transition matrix A is taken for further population generation. This process continues until we get the optimum value of P(O|λ).

The state transition matrix A is shown below. Each row of matrix A is considered as Chromosome. Each specific arrangement of these chromosomes is termed as a single population.

$A= \{a_{ij}\}$

	S1	S2	S3	S4	S5	S6	S7	S8
S1	0.1	0.2	0.2	0	0	0	0	0.5
S2	0.2	0.1	0.5	0	0	0	0	0.2
S3	0	0	0.1	0.4	0	0	0.4	0.1
S4	0	0	0	0.1	0.5	0.4	0	0
S5	0	0	0	0.6	0.1	0	0.3	0
S6	0	0	0	0.3	0.3	0.1	0.6	0
S7	0	0	0.15	0.15	0	0.3	0.1	0.3
S8	0.3	0.1	0.1	0.1	0	0	0.3	0.1

Table: 6.2.1 Initial state transition on probability matrix discipline chromosomes

Crossover: To generate random population we applied crossover operation on state transition matrix A. For that we have selected two random chromosomes (among 1 to 8) and interchange their position .State transition matrix A after interchanging chromosome 3 and 5:

	S1	S2	S3	S4	S5	S6	S7	S8
S1	0.1	0.2	0.2	0	0	0	0	0.5
S2	0.2	0.1	0.5	0	0	0	0	0.2
S3	0	0	0	0.6	0.1	0	0.3	0
S4	0	0	0	0.1	0.5	0.4	0	0
S5	0	0	0.1	0.4	0	0	0.4	0.1
S6	0	0	0	0.3	0.3	0.1	0.6	0
S7	0	0	0.15	0.15	0	0.3	0.1	0.3
S8	0.3	0.1	0.1	0.1	0	0	0.3	0.1

Table 6.2.2: Matrix A after applying Crossover operation

Mutation: - Mutation is applied on state transition matrix A by adding or subtracting a small number d in the range of (0.001- 0.009) from any randomly generated row and column. State transition matrix A after mutating the value A(3,5) by d=0.003

	S1	S2	S3	S4	S5	S6	S7	S8
S1	0.1	0.2	0.2	0	0	0	0	0.5
S2	0.2	0.1	0.497	0	0.003	0	0	0.2
S3	0	0	0.1	0.4	0	0	0.4	0.1
S4	0	0	0	0.1	0.5	0.4	0	0
S5	0	0	0	0.6	0.1	0	0.3	0
S6	0	0	0	0.3	0.3	0.1	0.6	0
S7	0	0	0.15	0.15	0	0.3	0.1	0.3
S8	0.3	0.1	0.1	0.1	0	0	0.3	0.1

Table 6.2.3: Matrix A after applying Mutation operation

6.3 Training using Genetic algorithms:

For training we have used the above described procedure until the optimum value of P(O|λ) is calculated. Table listed below shows the optimum values of P(O| λ) obtained after applying the above described procedure are on ten images of English character ***A***

Image Number	Optimum value of P(O| λ)	Value after subtracting the MAX value
1	-45.931374793766658	5.951343101964568
2	-45.024052893630682	5.044021201828592
3	-53.081643153039984	13.101611461237894
4	-46.293857621043280	6.313825929241190
5	-44.491400874619245	4.511369182817155
6	-46.841130028390438	6.861098336588348
7	-39.980031691802090	0
8	-46.456143959079796	6.476112267277706
9	-43.597508500817824	3.617476809015734
10	-43.961701605892614	3.981669914090524

Table 6.3.1 P(O|λ) values of ten images of character A by using Genetic Algorithms

Above result shows that range for English character ***A*** will be **0 to 13.101611.**

For testing, we input an image of new character and calculate the optimum value of P(O|λ). We subtracts this value from the maximum P(O|λ) value that we got during the training of that character, if the result lies in our already calculated range this implies that we have recognized character accurately, else our model fails to recognize that image correctly.

Table listed below shows the value of P (O|λ) calculated for six test images of English character ***A*** after subtracting them from the MAX value which was obtained during the time of training.

Test Image Number	Value of P(O|λ) after subtracting from the MAX value	Recognized
1	-1.750567630343475	No
2	3.970562730108057	Yes
3	7.262407602282607	Yes
4	2.282290161880660	Yes
5	7.483495368462940	Yes
6	8.002883980225448	Yes

Table 6.3.2 Results obtained by genetic algorithms

6.4 Analysis through forward, Baum-Welch and Genetic algorithms

Above result shows that five values out of six values lie in the range created for English character ***A***. It shows that our model have efficiency of 83.33% (approx.) in recognizing the English character **A.** The table listed below shows the efficiency of the all the three algorithms for the English alphabet **A**.

Algorithm	Efficiency for recognizing English Alphabet *A*
Forward Algorithm	99.90 % (approx.)
Baum-Welch Algorithm	66.67 %
Genetic Algorithm	83.33 %

Table 6.4.1 Comparison of various classification algorithms used in the paper

7 Conclusions

We have taken data of handwritten English characters (six different images of each character) and created Hidden Markov Model for that. We have applied three techniques for character recognition namely: Forward Algorithm, Baum-Welch Algorithm, and Genetic Algorithm. To applying these three techniques we can be conclude that our HMM Model (forward method) recognizes the handwritten characters approximately with the maximum efficiency. It can be shown by using the above result that Forward algorithm recognizes all the six images, whereas Baum-Welch Algorithm and Genetic Algorithm recognizes four and five character images of English character out of six character images respectively. Using these results we can apply this method to recognize the scanned image for characters in future use.

References

1. Lawrence R. Rabiner, IEEE, “A Tutorial on Hidden Markov Models and selected applications in Speech Recognition”.
2. Rajib Lochan Das, Binod Kumar Prasad, Goutam Sanyal “English Character Recognition using Global and Local Feature Extraction, International Journal of Computer Applications (0975 – 8887) Volume 46– No.10, May 2012.
3. 3.Somaya Alma’adeed, Colin Higgens, and Dave Elliman University of Nottingham, Recognition of Off-Line Handwritten Arabic Words Using Hidden Markov Model Approach
4. El-Yacoubi,R. Sabourin, M. GillouxC.Y. Suen “Off- Line Handwritten Word Recognition using Hidden Markov Model”.
5. Magdi A. Mohamed, Member, IEEE, and Paul Gader, “Generalized Hidden Markov Models—Part II: Application to Handwritten Word Recognition” Senior Member, IEEE, IEEE TRANSACTIONS ON FUZZY SYSTEMS, VOL. 8, NO. 1, FEBRUARY 2000
6. Binod Kumar Prasad**,** Goutam Sanyal**,** Department of Computer Science and Engineering, National Institute of Technology, Durgapur, INDIA, “ A Model Approach to Offline English Character Recognition” International Journal of Scientific and Research Publications, Volume 2, Issue 6, June 2012 1 ISSN 2250-3153
7. Jonathan J. Hull, Alan Commike and Tin-Kam Ho
8. J. J. Hull, S. N. Srihari, E. Cohen, C. L. Kuan, P. Cullen and P. Palumbo, "A blackboard-based approach to handwritten ZIP Code recognition, " International Conference on Pattern Recognition, Rome, Italy, November, 1988, 111-113.
9. D. Lee, S. W. Lam and S. N. Srihari, "A structural approach to recognize hand-printed and degraded machine-printed characters," submitted to the Symposium on Syntactic and Structural Pattern Recognition, Murray Hill, New Jersey, 1990.
10. J. J. Hull, "Inter-word constraints in visual word recognition," Proceedings of the Conference of the Canadian Society for

Computational Studies of Intelligence, Montreal, Canada, May 21-23, 1986, 134-138.

11. Nafiz Arica and Fatos T. Yarman-Vural
12. J. Serra, "Morphological filtering: An overview," Signal Process., vol. 38, no. 1, pp. 3–11, 1994.
13. M. Sonka, V. Hlavac, and R. Boyle, Image Processing, Analysis and Machine Vision, 2nd ed. Pacific Grove, CA: Brooks/Cole, 1999.
14. H. S. Baird, "Document image defect models," in Proc. Int. Workshop Syntactical Structural Pattern Recognit., 1990, pp. 38–47.
15. M. Cannon, J. Hockberg, and P. Kelly, "Quality assessment and restoration of typewritten document images," Int. J. Document Anal. Recognit., vol. 2, no. 2/3, pp. 80–89, 1999.
16. C. Downtown and C. G. Leedham, "Preprocessing and presorting of envelope images for automatic sorting using OCR," Pattern Recognit., vol. 23, no. 3–4, pp. 347–362, 1990.
17. W. Guerfaii and R. Plamondon, "Normalizing and restoring on-line handwriting," Pattern Recognit., vol. 26, no. 3, pp. 418–431, 1993.
18. L. O'Gorman, "The document spectrum for page layout analysis," IEEETrans. Pattern Anal. Machine Intell., vol. 15, pp. 162–173, 1993.
19. R. G. Casey and E. Lecolinet, "A survey of methods and strategies in character segmentation," IEEE Trans. Pattern Anal. Machine Intell., vol. 18, pp. 690–706, July 1996.
20. A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: A review," IEEE Trans. Pattern Anal. Machine Intell., vol. 22, pp. 4–38, Jan. 2000.
21. H. D. Block, B. W. Knight, and F. Rosenblatt, "Analysis of a four layer serious coupled perceptron," Rev. Mod. Phys., vol. 34, pp. 135–152,1962.
22. A. K. Jain and D. Zongker, "Representation and recognition of handwritten digits using deformable templates," IEEE Trans. Pattern Anal. Machine Intell., vol. 19, pp. 1386–1391, Dec. 1997.
23. M. Bokser, "Omnifont technologies," Proc. IEEE, vol. 80, pp. 1066–1078, 1992.
24. Dewi Nasien, Habibollah Haron, Siti Sophiayati Yuhaniz.
25. Suliman, A. Shakil, A. Sulaiman, M. N. Othman, M. and Wirza, R. "Hybrid of HMM and fuzzy logic for handwritten character recognition". In Proceedings of Information Technology. Kuala Lumpur, Malaysia, 2008
26. Zhaoqi, B. and Xuegong, Z. "Pattern Recognition", 2nd Edition, Tsinghua University Press.,(2000)
27. Ravindra Nath, and Renu Jain "Using Randomized Search Algorithms to Estimate HMM Learning Parameters" IEEE International Advanced computing Conference (IACC-2009).
28. A.H. Mantawy, L. Abdul Mazid, Z. Selim "Integrating genetic Algorithms, Tabu search and Simulated Annealing for the unit commitment problem", IEEE Transaction on power system, Vol.14, No. 3 August 1999.
29. Mark Pirlot, "general local search method", European journal of operational research -92 (1996) 493-511.