Duration Models for Arabic Text Recognition using Hidden Markov Models

Fouad SLIMANE[1,2], Rolf INGOLD[2], Adel M. ALIMI[1], Jean HENNEBERT[2,3]

[1]*REsearch Group on Intelligent Machines (REGIM)*
University of Sfax, National Engineering School of Sfax, BP. W-3038, Sfax, Tunisia
[2] *Department of Informatics*
University of Fribourg, Bd de Pérolles, 1700 Fribourg, Switzerland
[3] *Business Information Systems Institute, University of Applied Science Western Switzerland, HES-SO, TechnoArk 3, 3960 Sierre, Switzerland*
Fouad.Slimane@unifr.ch, Rolf.Ingold@unifr.ch, Adel.Alimi@enis.rnu.tn, Jean.Hennebert@unifr.ch

Abstract

We present in this paper a system for recognition of printed Arabic text based on Hidden Markov Models (HMM). While HMMs have been successfully used in the past for such a task, we report here on significant improvements of the recognition performance with the introduction of minimum and maximum duration models. The improvements allow us to build a system working in open vocabulary mode, i.e., without any limitations on the size of the vocabulary. The evaluation of our system is performed using HTK (Hidden Markov Model Toolkit) on a database of word images that are synthetically generated.

1. Introduction

Most of the optical text recognition systems for Latin printed text assume that the characters can be isolated with an a priori segmentation procedure. This phase of segmentation is difficult in the case of cursive or semi-cursive writing.

In the seventies, several solutions of Arabic printed texts recognition have already been proposed [14]. Systems for the recognition of Arabic script can be divided into two categories: the first one is based on a *global* approach (Global approach models the word in its entirety, indifferently of the composing characters) and the second one is based on an *analytical* approach (Analytical approach considers the word as a composition of sub-entities. These sub-entities correspond to a character in a given context or to a fragment of character).

However, the difficulty is now in the a priori segmentation of the word into these sub-entities. Several a priori segmentation approaches have been studied with limited success, see for example [3]. The recognition performance is actually very much dependent to the performance of the segmentation procedure.

To face the difficulties of varying characters shapes and of the segmentation procedure, several researchers have proposed to use Hidden Markov Models (HMMs) to recognize Arabic script [4], [5], [7], [8], [9], [10], [11], [12], [13], [14]. HMMs have been widely used in many tasks involving sequence modelling where the observations are emitted by distinct states. A wide literature is available on HMMs, especially in the field of speech recognition [15]. HMMs have also been applied to cursive handwriting recognition [3], [1].

HMMs, by nature, are modelling a double stochastic process: emission of observations and transition of states. In the case of Arabic script, a word is transformed into a sequence of feature vectors which are the observations input to the HMM. An HMM is then used to model the word where states are associated to characters, sub-characters or directly to their variations. The transition probabilities between states are typically modelling the probability that one character follows another one.

There are two main advantages of using HMMs to recognize Arabic script. First, the emission probability density functions of the HMMs can be trained to model variations of the character shapes. Second, the decoding procedure will solve in the same time the recognition of words and the segmentation into character models.

In this paper, we present several improvements of a state-of-the-art HMM system similar to the one reported in [7]. More specifically, we analyze the benefits of introducing HMM topologies that are able

978-0-7695-3514-2/08 $25.00 © 2008 IEEE
DOI 10.1109/CIMCA.2008.229

to model durations in characters[1]. Similar duration modelling approaches have already been introduced for the recognition of low-resolution text in images [6]. The improvements allow us to build an open vocabulary system, i.e., a system that can recognize any arbitrary word written in Arabic script, without limitations on the vocabulary size. The evaluation of our system is performed using the widely used Hidden Markov Model Toolkit (HTK) [16].

The paper is organized as follows. In Section 2, we present the general characteristics of the Arabic writing. In section 3, the recognition system is described in more details. Section 4 presents the evaluation framework used for this work. Results are discussed in Section 5 and are followed by our discussions.

2. General characteristics of the Arabic writing

As illustrated in Figure 1, the Arabic script is written from right to left and is semi-cursive either in printed form or handwritten. Each character, in a word, has a left and/or right connection point that, lies on an imaginary line, which is the base line. The writing is unicase (the concept of upper and lower case letters does not exist). The Arabic alphabet is richer than the Latin as it contains 28 letters that have different appearances according to their position in the word. Typically, the shape of one character can be different depending if it is isolated or if its position is at the beginning, middle or end of the word. For example, the letter "**خ** (Xaa)" has four kinds of appearances: isolated "**خ**" as in "**صرخ** (shouting)" at the beginning "**خـ**" as in "**خبرة** (experience)", in the middle "**ـخـ**" as in "مخبر (laboratory)" and at the end "**ـخ**" as in "**طبخ** (cooking)". We present in Table 1 the 28 Arabic letters with their different shapes according to their position in the word. Letters with just 2 shapes shown cannot be connected to the following letter, thus, their "Begin" shapes are simply their "Isolated" shapes, and their "Middle" shapes are their "End" shapes.

Arabic word can be composed of one or more components (pseudo-word) and the characters of the same connected component can be ligatured horizontally and vertically. The ligature procedure is also dependent to the font used (in some fonts, we can go up to four vertical ligatures). Figure 1 shows a vertical ligature of three characters Alif, Miim and Jiim.

Table 1. Arabic letters

Letter label	Isolated	Begin	Middle	End
Alif	ا		ـا	
Baa	ب	بـ	ـبـ	ـب
Taaa	ت	تـ	ـتـ	ـت
Thaa	ث	ثـ	ـثـ	ـث
Jiim	ج	جـ	ـجـ	ـج
Haaa	ح	حـ	ـحـ	ـح
Xaa	خ	خـ	ـخـ	ـخ
Daal	د		ـد	
Thaal	ذ		ـذ	
Raa	ر		ـر	
Zaay	ز		ـز	
Siin	س	سـ	ـسـ	ـس
Shiin	ش	شـ	ـشـ	ـش
Saad	ص	صـ	ـصـ	ـص
Daad	ض	ضـ	ـضـ	ـض
Thaaa	ط	طـ	ـطـ	ـط
Taa	ظ	ظـ	ـظـ	ـظ
Ayn	ع	عـ	ـعـ	ـع
Ghayn	غ	غـ	ـغـ	ـغ
Faa	ف	فـ	ـفـ	ـف
Gaaf	ق	قـ	ـقـ	ـق
Kaaf	ك	كـ	ـكـ	ـك
Laam	ل	لـ	ـلـ	ـل
Miim	م	مـ	ـمـ	ـم
Nuun	ن	نـ	ـنـ	ـن
Haa	ه	هـ	ـهـ	ـه
Waaw	و		ـو	
Yaa	ي	يـ	ـيـ	ـي

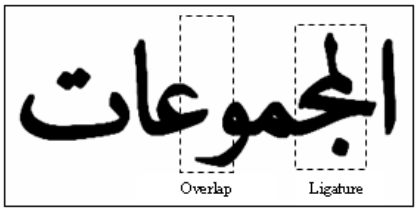

Figure 1. Example of Arabic word with a vertical ligature and a vertical overlap

Finally, vertical overlaps can occur at the intersection of pseudo-words and also within words for some sequence of characters. For more details on the characteristics of the Arabic script, we refer to [14].

3. System description

The proposed system is based on HTK. As illustrated in Figure 2, the system is working in two

[1] Another difference in comparison to [7] is also the introduction of continuous estimators for the emission probability density functions, instead of discrete estimators.

phases: training and recognition. These two phases are sharing the same feature extraction frontend.

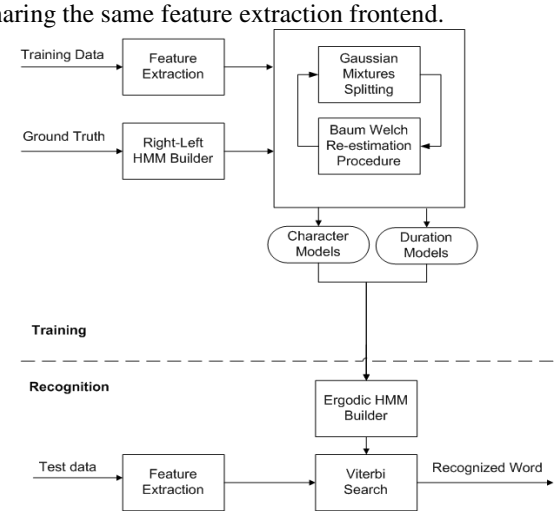

Figure 2. Processing steps involved in the HTK-based Arabic recognition system

3.1. Preprocessing and feature extraction

In the feature extraction part, words of each image are transformed into a sequence of feature vectors computed from a narrow analysis window of *N* pixels sliding from right to left on the word. In our settings, the uniform analysis window is shifted of 1 pixel. We performed several tests to determine the optimal size of the window to maximize the recognition rate and we obtained an optimal value with *N=14*. The size of this analysis window is of course dependent on the font size and font family and could be different for another task. A feature vector is extracted from each analysis window. As a result, no segmentation into letters is made and the word is transformed into a matrix of values where the number of lines corresponds to the number of analysis windows and the number of columns is equal to the number of coefficients in each feature vector.

The feature extraction is divided into two parts. The first part extracts, for each window:

- the number N_1 of black connected components
- the number N_2 of white connected components
- the ratio N_1/N_2 of black and white connected components
- the position of the smallest black connected component divided by the height of the window
- the perimeter of all components in window/perimeter of window
- compactness $\left((Perimeter)^2/(4\pi \text{Area})\right)$
- gravity centre of the window, of the right and left half and of the first third, the second and the last part of the window:

$$G_x = \frac{\sum_{i=1}^{n} x_i}{n \times Width} \quad G_y = \frac{\sum_{i=1}^{n} y_i}{n \times Height}$$

The second part of the feature extraction consists of resizing the window into a normalized size of 20 pixels height and computing the horizontal and vertical projection values. The feature extraction, overall, results in a vector of 53 coefficients. Our feature extraction module is here implemented in Java.

3.2. HMM with HTK

We used the Hidden Markov model Toolkit (HTK) to realize our evaluation [16]. HTK was originally developed at the Speech Vision and Robotics Group of the Cambridge University Engineering Department (CUED). This toolbox has been built to experiments with Hidden Markov Models (HMMs) and has been extensively used in speech recognition research. HTK is a set of command line executables used for initializing, modifying, training and testing HMMs. The use of HTK typically goes through four phases: preparation of data, training, recognition and recognition performance evaluation.

In the preparation phase of data, the sequences of features vectors representing the words are extracted as described in Section 3.1 and are converted into a file format compatible with HTK using the HTK HCopy command.

In the learning phase, all training files are firstly used for the initialisation of HMM models for each letters, using HTK HCompV. For each training word image, the corresponding sub-models are connected together to form a *right-left* HMM. We experimented in this paper using different sub-model topologies as explained below in Section 3.3, the simplest topology being the single-state one as illustrated on Figure 3.B. An embedded training using the Baum-Welch iterative estimation procedure is used with HTK tool HERest.

Using training set, all the observation sequences are used to estimate the emission probability functions of each sub-model. The training procedure actually involves two steps that are iteratively applied to increase the number of Gaussian mixtures to a given *M* value. In the first step, a binary split procedure is applied to the Gaussians to increase their number. In the second step, the Baum-Welch re-estimation procedure is launched to estimate the parameters of the Gaussians.

At the end of the training phase, when applied, duration models are derived from a forced alignment procedure applied to the training data. The duration values are then applied to alter the HMM topology as explained below.

At recognition time, an ergodic HMM is composed using all sub-models. All transitions from one sub-model to the other are allowed. The proposed model allows recognizing potentially any word in an open vocabulary fashion. The disadvantage of this procedure is that the system can propose invalid words as recognition. However, we show through the results obtained in this paper that the use of duration modelling at recognition time is reducing drastically the recognition errors of the system. Using ergodic topology offers the advantage of relatively lightweight memory and cpu footprint, when compared to more heavyweight approaches based on finite-state or stochastic grammars. The recognition is done by looking the best state sequence in the HMM using a Viterbi procedure implemented with the HTK tool HVite.

Performances are evaluated in terms of word recognition rates using an unseen set of word images. It is obtained using the HTK HResult tool.

3.3. Sub-model topologies

The proposed recognition model is ergodic where all sub-models are connected together. Each model represents a letter or several letters (in the case of letters ligatures). Table 2 below summarizes the 60 different sub-models that have been selected for our system. The selection procedure of the different sub-models has been driven by grouping shapes of letters presenting few variations. Our assumption is here that the emission probability estimators based on Gaussian mixtures will offer enough flexibility to model the common parts and the variations within each letter category.

Using the terminology introduced for speech recognition [15], our models are said to be context independent, i.e., each sub-model is considered independent to the next.

We explored in this paper different context independent sub-model topologies that we present in three categories:

1. **Equal number of states for each model**: an a priori fixed HMM topology is the same for all models, for example 1 state per sub-model (Figure 3.B) or 3 states per sub-model (Figure 3.C).
2. **Number of states dependent to relative width and form of letters**: the number of states in the sub-model is dependent with form and with of letter image, for example *alif_E* is set to 1 state while *saad_I* is set to 5 states.
3. **Duration models based**: a simple single-state HMM topology is used during training and is altered during testing to include a minimum duration model that is either knowledge-based using font metric information or inferred from the duration values obtained during training. The minimum duration model is simply obtained by repeating a given state D_i times to *force* the decoding process to spend a minimum of feature vectors in the sub-model i.

Table 2: Sub-models labels and their corresponding letter(s).

Label	Letters	Label	Letters
Baa_B	بـ ب	TaaaClosed_E	ة ـة
Taaa_M	تـ ت	YaaChadda_I	يّـ ـيّ
Raa_I	ر ـر	YaaChadda_M	ـيّـ
Ghayn_B	غـ	Hamza	ء
Alif_E	ـا ا	Laam_I	لـ ل
Nuun_B	نـ ن	Laam_M	ـلـ ـل
Taaa_E	ت ـت	LaamAlif_I	لا لا
Miim_B	مـ م	Thaa_M	ـث ـث
Miim_I	م ـم	Gaaf_I	قـ ق
Thaaa_I	ط ـط ـطـ ط	Gaaf_B	قـ ـقـ
Waaw	و ـو	Jiim_M	ـجـ جـ
Nuun_I	ن ـن	Haaa_B	حـ ـحـ
Yaa_M	يـ ـيـ	Thaa_E	ث ـث
Yaa_I	ي ـي	HamzaUnderAlif_I	إ ـإ
Saad_I	ص ـص	HaaChadda_E	ـهّ
Xaa_B	خـ خ	HamzaAboveAlif	أ ـأ آ ـآ
Siin_I	س ـس	NuunChadda_E	ـنّ نّ
Siin_B	سـ ـسـ	Shiin_M	ـشـ شـ
Daal_I	د ـد	Taa_I	ظ ـظ ـظـ ظ
Haa_I	ه	Baa_E	ـب ب
Haa_B	هـ ـهـ	Shiin_I	ش ـش
Thaal_I	ذ ـذ	Saad_B	صـ ـصـ
Xaa_I	خ ـخ	Ayn_M	ـعـ
AlifBroken_E	ى ـى	Ayn_B	عـ
Haaa_I	ح ـح	Ghayn_M	ـغـ
Zaay_I	ز ـز	Daad_I	ض ـض
Kaaf_I	ك ـك	Daad_M	ضـ ـضـ
Kaaf_B	كـ ـكـ	Jiim_I	ج ـج
Ghayn_I	غ	Ghayn_E	ـغ
Ayn_E	ع	Faa_M	ـفـ ـف

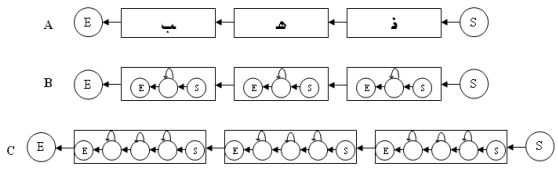

Figure 3. Example of topologies using a fixed and equal number of states for sub-models (B:1-state)(C:3-states)

4. Evaluation Framework

For the evaluation of our system, we have used a synthetic database of word images composed of 20'630 word images. We generated the image using a Java procedure using the font *Times,* 24 points. The resulting images have a size of 56 pixels height as a maximum and are stored in PNG format. The generated images are synthetic but are somehow similar to what can be seen in standard newspapers.

Using a synthetic database presents advantages and disadvantages. On the one side, the synthetic approach allows us to generate a quite large database of words images where the words are already isolated. The large quantity of data is beneficial in terms of evaluation as we can let the HMM training procedure converge to precise values of the probability density functions estimators. On the other side, the database does not contain variabilities that are usually present when scanning a document.

Our objective in this paper is to evaluate the impact of duration models in a HMM system. The synthetic database used in this work is therefore sufficient for our objectives.

The evaluation database is divided into a training set of 19'630 different and into a test set of 1'000 words. These sets are fully independent.

5. The experimental results

All results (grouped in Table 3) were obtained keeping constant the different feature extraction parameters (*N=14* pixels, shift of 1 pixel), model complexity (64 mixtures in each states) and training procedure (same number of iterations, same algorithm parameters).

Results obtained using an equal number of states for each model show increasingly good results when the number of states per sub-models is increasing from 1 to 6 states. The best results are obtained with 6 states per sub-models with 93.1% of word recognition rate.

Several comments can be given on these results:

- While performances are increased when model topologies have more states, this is at the cost of recognition time and memory footprints.
- Although putting more states in each model seems to increase the performances, we could not go above 6 states per sub-models as training convergence conditions were not anymore met with HTK.
- The results in terms of word recognition rates are low for single state models. However, the character recognition rates were mesure to be quite high at 99.4%. This means that most of the letters are well recognized while some letters are inserted or deleted in many words. We did this observation for example with letter *alif_E* which is actually similar to sub-parts of many of the other letters and was then inserted frequently in words.

Table 3: Results system using a fixed number of states

Categ.	Train Model	Test Model	% Word
1	1-state	1-state	7.0
1	3-states	3-states	70.4
1	5-states	5-states	92.9
1	6-states	6-states	93.1
2	[1,3]-states	[1,3]-states	18.1
2	[3,5]-states	[3,5]-states	85.9
2	[1,3,5]-states	[1,3,5]-states	78.5
2	[3,5,6]-states	[3,5,6]-states	88.6
3	1-state	knowledge-based min-duration	81.5
3	1-state	min-duration $d=\mu_d/2$	91.9

In the second category of model topologies, the number of states is dependent on the relative length of each letter. For example, in the configuration [1,3,5]-states, we have used 1 state for short letters, 3 states for medium sized letters and 5 states for long sized letters. The affectation of a sub-model to a given length was performed manually, inspecting their relative size. The motivations are here to provide richer models for longer letters while keeping low cpu and memory footprint by affecting less demanding models for the shortest states. The results show that the performances are always in between the performances of the configurations of category 1.

For minimum duration models, we report about two experiments. For the first one, the minimum duration values are knowledge-based, measured manually from all letters and injected in the system. For the second experiment, the minimum duration values D_i for each sub-model i are automatically inferred at training time, performing a Viterbi forced alignment on the input images and accumulating in histograms the number of times the self-loop transition is visited for all states. The average number of self-loop transitions μ_i can be simply computed from the histograms. In our experiments, the minimum duration value D_i has been set to half of this average value $D_i = \mu_i/2$ [15]. The following observations can be done:

- Inferring the minimum duration values from training time leads to better recognition results than the knowledge base approach. This is probably due to the segmentation procedures that are different between the manual and automatic approach.
- When comparing the single state results (7%) and the minimum duration results (91.9%), we can observe all the benefits of introducing such minimum durations. The memory footprint remains more or less

exactly the same (60 bytes more to encode the duration values) and the cpu usage also show benefits thanks to a better pruning of less probable paths in the decoding procedure.

The most interesting observation from these experiments is that we can recover similar performances of more complex models (6-states configuration) using less cpu and memory consuming single-state models by injecting simple minimum duration constraints in the HMM topologies.

A remaining problem with the single state configuration is the systematic mis-recognition of words where same letters are consecutives. It is indeed equivalent for the system to remain more in a given state than to emit two similar consecutive states. This drawback will be addressed in future work with the introduction of other duration models that take into account minimum and maximum constraints.

6. Conclusion

In this paper, we have presented a system for open-vocabulary recognition of Arabic printed text based on HMMs. The benefit of HMMs is clearly in its ability to segment the semi-cursive Arabic script into letters while performing the recognition at the same time. The novelty of the work reported in this paper over the state of the art in Arabic recognition is in the introduction of minimum duration models that allow to increase significantly the performances of the system while keeping an architecture which is lightweight in terms of cpu and memory.

7. References

[1]A. Schlapbach and H. Bunke, “Using HMM-based recognizers for writer identification and verification”, Proc. 9th Int. Workshop on Frontiers in Handwriting Recognition, 2004, pp. 167-172.

[2] B. Al-Badr, S.A. Mahmoud, “Survey and bibliography of Arabic Optical Text Recognition”, Signal processing, 1995, vol. 41, pp. 49-77.

[3]U-V. Marti and H. Bunke, “Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition system”, Journal of Pattern Recognition and Art. Intelligence 15, 2001, pp. 65-90.

[4] H. Miled, C. Olivier, M. Cheriet, Y. Lecourtier, “Coupling observation/letter for a Markovian modelisation applied to the recognition of arabic handwriting”, IEEE Proc. 4th International conference on document analysis and recognition (ICDAR'97), Ulm, Germany, 1997, pp. 580-583.

[5] H. Miled, M. Cheriet, C. Olivier, Y. Lecourtier, “Modélisation markovienne de l'écriture arabe manuscrite: une approche analytique”, Proc. 1er Colloque international francophone sur l'écrit et le document (CIFED'98), Québec, Canada, 1998, pp. 50-59.

[6] F. Einsele, R. Ingold, J. Hennebert, "A Language-Independent, Open-Vocabulary System Based on HMMs for Recognition of Ultra Low Resolution Words", In proc. of 23rd Annual ACM Symposium on Applied Computing (ACM SAC 2008), Fortaleza (Brasil), March 16 - 21 2008.

[7] M S. Khorsheed, “Offline recognition of omnifont Arabic text using the HMM ToolKit (HTK)”, Pattern Recognition Letters 28(12), 2007, pp. 1563-1571.

[8] M. S. Khorsheed, “Off-line arabic character recognition–– a review”, Pattern Anal, 2002, Appl. 5, 31–45.

[9] M. S. Khorsheed, Clocksin, W.F., “Multi-font arabic word recognition using spectral features”, The 15th International Conference on Pattern Recognition ICPR, vol. 4. Barcelona, Spain, 2000, pp. 543–546.

[10] M. S. Khorsheed, “Recognising handwritten Arabic manuscripts using a single hidden Markov model”, Pattern Recognition Letters, v.24 n.14, October 2003, pp.2235-2242.

[11] M.C. Fehri, “Reconnaissance de textes arabes mutifonte à l'aide d'une approche hybride neuro-markoviennes”. Thèse de doctorat, Université des sciences, des techniques et de médecine de Tunis II, Tunisie, 1999.

[12] M.C. Fehri, M. Ben Ahmed, “Off-line arabic handwriting recognition”, Computational engineering in systems applications (CESA'98), Nabeul-Hammamet, Tunisie, 1998, pp.1-3.

[13] N. Ben Amara : “Utilisation des modèles de Markov cachés planaires en reconnaissance de l'écriture arabe imprimée”. Thèse de doctorat, Université des sciences, des techniques et de médecine de Tunis II, Tunisie, 1999.

[14] N. Ben Amara, A. Belaïd and N. Ellouze, “Utilisation des modèles markoviens en reconnaissance de l'écriture arabe :État de l’art”, CIFED, 2000.

[15] Rabiner, L., Juang, B., “Fundamentals of Speech Recognition”, Prentice Hall, 1993.

[16] Young, S., Evermann, G., Kershaw, D., Moore, D., Odell, J., Ollason, D., Valtchev, V., Woodland, P., “The HTK Book”, Cambridge University Engineering Dept., 2001.