JOURNAL OF ENGINEERING AND APPLIED SCIENCE, VOL. 54, NO. 6, DEC. 2007, PP. 653-672
FACULTY OF ENGINEERING, CAIRO UNIVERSITY

ARABIC OCR SYSTEM ANALOGOUS TO HMM-BASED ASR SYSTEMS; IMPLEMENTATION AND EVALUATION

M. A. RASHWAN[1], M. W. FAKHR[2], M. ATTIA[3], AND M. S. EL-MAHALLAWY[4]

ABSTRACT

Despite 35 years of R&D on the problem of Optical character Recognition (OCR), the technology is not yet mature enough for the Arabic font-written script compared with Latin-based ones. There is still a wide room for enhancements as per: lowering the Word Error Rate "WER", standing robust in face of moderate noise, and working on an omni-font open-vocabulary basis. Among the best trials done in this regard so far comes the HMM-based ones. Elaborating on this Automatic Speech Recognition "ASR"-inspired promising approach, our team has significantly refined basic processes and modules deployed in such architectures "e.g. lines & words decomposition, features extraction, models parameters selection, language modelling, .., etc." to develop what is hoped to be a truly reliable "i.e. low WER, omni font-written, open-vocabulary, noise-robust, and responsive" Arabic OCR suitable for real-life IT applications. This paper extensively reviews the HMM-based approach for building Arabic font-written OCR's in general, and our work in specific. It also reports about the experimental results obtained so far showing that our system outperforms its rivals reported in the published literature.

KEYWORDS: Arabic, ASR, characters, font-written, grapheme, HMM, ligatures, LBG, MAP, omni, pattern recognition, stochastic, SLM, speech.

1. INTRODUCTION

OCR is the branch of pattern recognition that ultimately aims to compete with the human ability to read written/printed text regarding both the speed and accuracy by associating character codes "e.g. ASCII codes" to digitally acquired images of characters "i.e. graphemes" [1].

[1] Prof. of Electronics and Communications Dept., Fac. of Eng., Cairo University, Giza, Egypt.
[2] Dean of the faculty of Computers & Informatics, Arab Academy for Science & Technology and Maritime Transport; (AASTMT) , Egypt,
[3] Human Language Technologies (HLT) consultant, the Eng. Company for the Development of Computer Systems; RDI.
[4] Teaching assistant of Electronics and Communications, Arab Academy of Science & Technology and Maritime Transport; (AASTMT) , Egypt

The potential application areas of automatic reading machines are numerous [1-4]. The text produced by OCRing text images can be used for all kinds of Information Retrieval “IR” and Knowledge Management “KM” systems which are not so sensitive to the inevitable Word Error Rate “WER” of whatever OCR system as long as this WER is kept lower than 10% to 15% [5]. It might be sufficiently illustrative to mention the gigantic project of the online global cross lingual searchable library being achieved by Google ™ as an example of how OCR is vital in this regard [6].

Although more than half a billion people worldwide, with various oriental languages, use the Arabic script for writing “alongside Arabic, Persian and Urdu are the most noted examples”, Arabic OCR technology is well behind its counterparts for other languages especially those based on the Latin script. The best recently reported Arabic omni font-written OCR systems can only claim WER ’s exceeding 10% under favorable conditions and not to mention the real-life ones [7]. This may be attributed mostly to the special characteristics of Arabic script detailed in the following section.

2. ARABIC OCR CHALLENGES [1, 8]

2.1 The Connectivity Challenge

Whether handwritten or font-written, Arabic text can only be scripted cursively; i.e. graphemes are connected to one another within the same word with this connection interrupted at few certain characters or at the end of the word as shown in Fig. 1. This necessitates any Arabic OCR to do not only the separate graphemes recognition task, but also another may be tougher graphemes segmentation one. To make things even harder, both of these tasks are mutually dependent and must hence be done simultaneously.

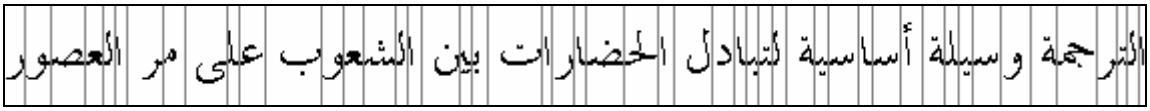

Fig. 1. Graphemes segmentation process illustrated by manually inserting vertical lines at the appropriate grapheme connection points.

2.2 The Dotting Challenge

Dotting is extensively used to differentiate characters sharing similar graphemes. According to Fig. 2 below, where some example sets of dotting-differentiated graphemes, it is apparent that the digital differences between the members of the same set are small. Whether the dots are eliminated before the recognition process, or recognition features are extracted from the dotted script, dotting is a significant source of confusion – hence recognition errors – in Arabic OCR systems especially when run on noisy documents; e.g. those reproduced by photocopiers.

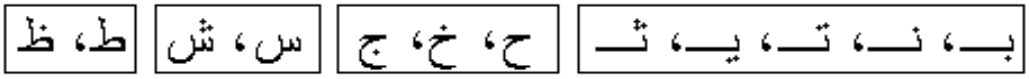

Fig. 2: Example sets of dotting-differentiated graphemes.

2.3 The Multiple Grapheme Cases Challenge

Due to the mandatory connectivity in Arabic orthography; the same grapheme representing the same character can have multiple variants according to its relative position within the Arabic word segment {Starting, Middle, Ending, Separate} as exemplified by the 4 variants of the Arabic character “Ein” shown in bold in Fig. 3.

Fig. 3. Grapheme “Ein” in its 4 positions; Starting, Middle, Ending & Separate.

2.4 The Ligatures Challenge

To make things even more complex, certain compounds of characters at certain positions of the Arabic word segments are represented by single atomic graphemes called ligatures. Ligatures are found in almost all the Arabic fonts, but their number depends on the involvement of the specific font in use. Traditional Arabic for example contains around 220 graphemes, and another common less involved font “with fewer ligatures” like Simplified Arabic contains around 151 graphemes. Compare this to English where 40 or 50 graphemes are enough. Again, a broader graphemes set means

higher ambiguity for the same recognition methodology, and hence more confusion [9]. Fig. 4 illustrates some ligatures in the famous font "Traditional Arabic".

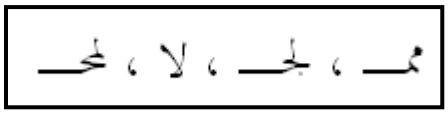

Fig. 4. Some ligatures in the Traditional Arabic font.

2.5 The Diacritics Challenge

Arabic diacritics are used in practice only when they help in resolving linguistic ambiguity of the text. The problem of diacritics with Arabic OCR is that their direction of flow is vertical while the main writing direction of the body Arabic text is horizontal from right to left as shown in Fig. 5. Like dots; diacritics – when existent - are a source of confusion of font written OCR systems especially when run on noisy documents, but due to their relatively larger size they are usually preprocessed [4].

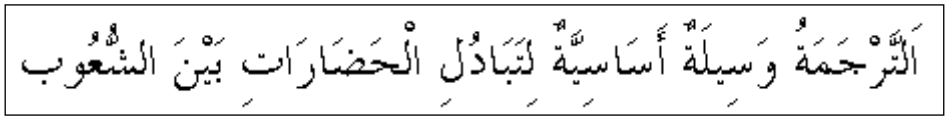

Fig. 5. Arabic text with diacritics.

2.6 The Size Variation Challenge

Different Arabic graphemes do not have a fixed height nor a fixed width. Moreover, neither the different nominal sizes of the same font scale linearly with their actual line heights, nor the different fonts with the same nominal size have a fixed line height.

3. ASR AND OCR PROBLEM ANALOGY

Solutions that tackle ASR and font written OCR are typically based on the concept of sub-word units "phonemes and graphemes" that are concatenated to form utterances and words respectively.

In ASR; the speech signal, which is a 1D function of time, is sliced into a sequence of windows "called frames" as shown in Fig. 6 and a features vector for each frame is computed. Frame widths should not be too wide nor too narrow. Were the

former case holds true, a large portion of the computed feature vectors would be taken from inter-phoneme regions leading to the inability to neither build stable models nor recognize the basic units. Were the latter condition holds true, the computed feature vectors would be unable to reflect the local characterizing properties of the signal.

As the phoneme widths typically range between 40-to-250 ms, and as the digital acquisition of speech signals are done at about 44 k bits/s at 256 PCM, a compromise is easily made in speech and frame widths are typically chosen to be within 5-to-10 ms where there are enough samples to reflect the local nature of the signal. In the world of digital speech processing, Mel Frequency Cepstral Coefficients "MFCCs" [10] are the well established and widely agreed-upon features to characterize the local identity of the speech signal.

On the other hand, the OCR problem is a one of recognizing a 2D text image signal, with the sliding window moving on the writing direction "from right-to-left in the case of Arabic as shown in Fig. 6" and having a fixed height equal to the one of the word/line being analyzed.

To keep HMM search lattices at reasonable widths, which in turn reduces the computational complexity hence the recognition WER [11], the word "not the line" has been chosen in our work as the major unit for training and recognition. Using an enhanced histogram-based approach [12], the text image is therefore decomposed first into horizontal lines, and the lines are in turn decomposed into words as explained in Section 4.3 later.

Selecting the frame width is subject to the same compromise mentioned just above while talking about speech frames. Taking into consideration the typical printing resolution of 1200 dpi, the minimum scanning resolution of 300 dpi, and a smallest font size of 10, the width of the slimmest ligature at that font size is 4 pixels. This puts the limit of the frame width that should not be exceeded.

Despite the existence of well crafted image characterizers that are proven effective in discriminating images as a whole [13, 14], none of them is very successful to describe text images differentially through a sequence of narrow frames inside the sliding window. Here comes the main challenge of applying the OCR-to-ASR analogy

that the absence of a standard features set that are able to locally identify text-image signals in a way that reveals the essential writing concepts so that the smallest variance possible is preserved over the feature vectors obtained from different fonts/sizes of the same sequence of graphemes.

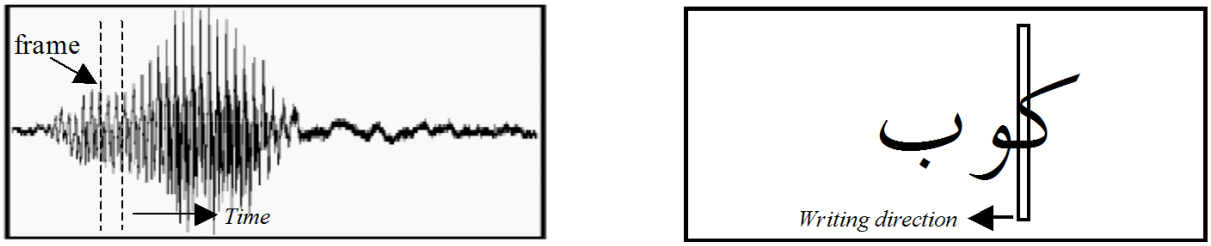

Fig. 6. Sliding window over a speech signal and over a text image bitmap.

Given the OCR-to-ASR analogy, the widely used approach to solving the ASR problem is then intuitively quoted for solving the Arabic font-written OCR one.

The writing process, like speech and like any other form of language, starts in the mind of some human writer when he/she composes a message to be delivered to his/her intended reader"s". The ultimate goal of this process is to recognize the message as intended by analyzing the message as observed. This is illustrated by the famous noisy channel communication model shown below.

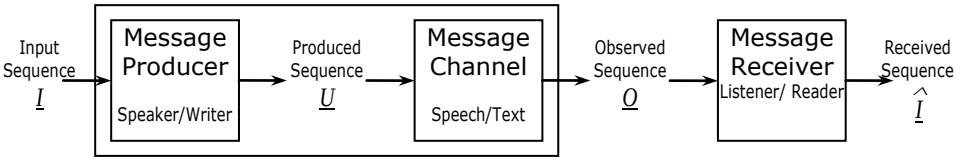

Fig.7. Noisy channel communication model where $\underline{I}$ is the intended message composed by the producer, and $\underline{O}$ is the message as observed by the receiver.

The distortion caused by the noisy channel typically leaves us with the ambiguity problem of having multiple possible perceived sequences from which we have to recognize the intended message sequence. While building a fully rule-based language recognizer is almost impossible due to the randomistic nature of the channel distortion, as well as our typical incomplete knowledge of the laws that govern the linguistic phenomenon, the goal of a language recognizer-like OCR-of finding the

intended sequence-of graphemes/ligatures in the case of OCR-can practically be achieved stochastically [15].

The maximum a posteriori probability "MAP" approach is one of the most widely-used and mathematically well-founded methodologies in this regard. According to this methodology, the elected message sequence $\hat{\underline{I}}$ is selected to maximize the a posteriori probability $P(\underline{I}|\underline{O})$ over all the possible "producible" messages which is formally expressed as,

$$\hat{\underline{I}} = \arg\max_{\forall \underline{I}} \{P(\underline{I} \mid \underline{O})\} = \arg\max_{\forall \underline{I}} \left\{ \frac{P(\underline{O} \mid \underline{I}) \cdot P(\underline{I})}{P(\underline{O})} \right\} = \arg\max_{\forall \underline{I}} \{P(\underline{O} \mid \underline{I}) \cdot P(\underline{I})\} \qquad (1)$$

$P(\underline{O} \mid \underline{I})$ is called the likelihood probability modeling the forward conditional stochastic relation between intended/input ligatures and their consequent observations, and $P(\underline{I})$ is called the language model that gives the a priori marginal probability of any possible sequence of ligatures. The a priori probability of the observations $P(\underline{O})$ can obviously be omitted from the maximization formula as it is independent of $\underline{I}$ [15].

In our OCR system-like the ASR one - the likelihood probability term $P(\underline{O} \mid \underline{I})$ is modeled via the public widely-used stochastic methodology of hidden Markov models "HMM" that has dominated the world of digital speech processing–esp. ASR-during the past two decades. The success of HMM in this regard is attracting researchers to deploy it as a suitable tool for cursive script recognition for the following reasons: [11]

- HMM 's are stochastic models that can cope with noise, and pattern variations occurring due to different fonts, sizes, and styles.
- The number of observations representing an unknown input grapheme/ligature or word may be of variable length, which is a fundamental requirement in cursive script as the lengths of individual units may greatly vary.
- The HMM recognition process implemented via search trellis does not only produce the MAP optimal sequence, but also the boundaries of the units comprising this sequence. In Arabic font-written OCR, this eliminates the need for any heuristic separate ligatures segmentation processes which are typically not only difficult but also error-prone.

- There are rigorously-studied, stable, and computationally efficient algorithms for the training, the adaptation, and the recognition using HMM 's. Moreover. Off-the-shelf implementations of such algorithms are easily affordable.

4. HMM-BASED OCR SYSTEM ARCHITECTURE

Based on the aforementioned OCR-to-ASR analogy, the simplified HMM-based solution architecture of the ASR problem shown in Fig. 8 below can be adopted to produce an effective solution for the Arabic font-written OCR problem whose architecture is illustrated by Fig. 9 below.

Our HMM-based Arabic OCR system illustrated in Fig. 9 above is composed of two phases: the initialization & training phase whose paths are drawn in dotted lines, and the recognition phase whose paths are drawn in solid lines.

The modules of this system are implemented using a variety of tools; MATLAB, HTK toolkit [16], as well as programming in C++. These modules are discussed further in the following subsections.

4.1 Digital Scanning

The scanning resolution used is preferred to be 600 dpi as the laser printing quality nowadays is at least 600 dpi. Also, this resolution is demanded for the slim characters of small sizes of compact fonts to have enough features to cope with the number of states of the HMM used. Scanning at a higher resolution may be desirable but not practical as it would need too high storage space "one A4 page scanned at B&W 1200 dpi would need more than 16M Bytes" and would moreover need more processing time during feature extraction and recognition phases. As demanded by our feature extractor explained in section. 4.4 later, input text images to our system should be bi-level "i.e. Black & White" which is performed via thresholding by the scanner hardware to save storage and time.

4.2 Primary Noise Handler

The target text rectangle bitmap is passed through a median filter in order to clean the salt & pepper noise typically occurring in noisy images extracted, for

example, from photocopied documents. In our study, a 5×5 median filter has proved effective for cleaning input scanned text images [13, 14].

Median filters are, however, hard deciding and cause considerable erosive distortion to the graphemes of written text which is generally harmful to the recognition performance. So, they should be used only during the lines & words decomposition, whereas other softer deciding filters–like Gaussian ones–might be used- f necessary-during the recognition process.

4.3 Words & Lines Decomposer

OCR systems need to perform a vital processing on the input scanned text rectangle prior to the recognition phase for decomposing the text rectangle into lines hence into words as illustrated in Fig. 10 and Fig. 11. Errors while this preprocessing add in the total error rate of the whole OCR system. This can be simply shown by

$$\begin{aligned} & a = 1 - e = (1 - e_D) \cdot (1 - e_R) = 1 - (e_D + e_R - e_D \cdot e_R) \\ & \because e_D \cdot e_R << 1 \\ & \therefore e \approx e_D + e_R \end{aligned} \tag{2}$$

where e_D is the lines & words decomposition error rate, e_R is the recognition WER, and e is the overall system WER. It is hence obvious how critical is the reliability of the lines & words decomposer to the viability of the whole OCR system.

One of the contributions of the work presented in this paper is developing an enhanced histogram-based algorithm [12] for this kind of decomposition so that it performs robustly on documents containing the idiosyncrasies of real-life documents especially as per noise and structural textual complexities

4.4 Features Extraction Module

Features extraction is one of the most important factors in achieving high recognition performance in pattern recognition systems. Our features extractor computes a features vector for each frame while sliding through each Arabic word from right to left producing a series of feature vectors injected to the vector quantizer.

Despite the existence of several image characterization methods [17], most of them can only succeed in integrally identifying an image as a whole. Invariant

moments [13] is a famous example that may be good for globally characterizing characters as a whole, so it can be effectively used in isolated characters recognition. However, applying this method on the narrow differential frames over words does not lead to omni-font open-vocabulary Arabic OCR systems.

In the published literature on HMM-based Arabic font-written OCR, the underlying features are merely simple time-domain luminosity coded features of the frames inside the sliding window as those described by [4], [7], and [18]. Unfortunately, the formulation of those features are ad. hoc. with limited success in building truly omni font-written Arabic OCR.

One major contribution of the work presented in this paper is the design of a mathematically rigorous lossless differential luminosity coding based features. Given a sequence of such features, one can fully retrieve the shape of the input written script with only a scaling factor as well as an inevitable digital distortion error.

Each single-pixel frame of a given Arabic word has a limited maximum number of vertical dark segments "4 segments are found to be enough". Each of these segments is coded into 4 components differentially representing the topological and agglomerative properties of the segment as follows,

i. The segment's center of gravity with respect to that of the previous frame.

ii. The segment's length with respect to that of the previous frame.

iii, iv. The orders of the most bottom, and the top segments in the preceding frame which are in 8-connection with the current segment. Special negative codes are given in case of non-connected segments.

Empty segments are padded by zeros. The dimensionality of our features vector is 16 = 4 segments × 4 components/segment. Our features vector components comprise both continuously analog as well as quantized values. Upon the examination of large populations of these feature vectors, it got clear that it is hard to find mixtures of Gaussians that properly represent that kind of hybrid data. So, the decision is made to adopt discrete HMM's rather than its other versions.

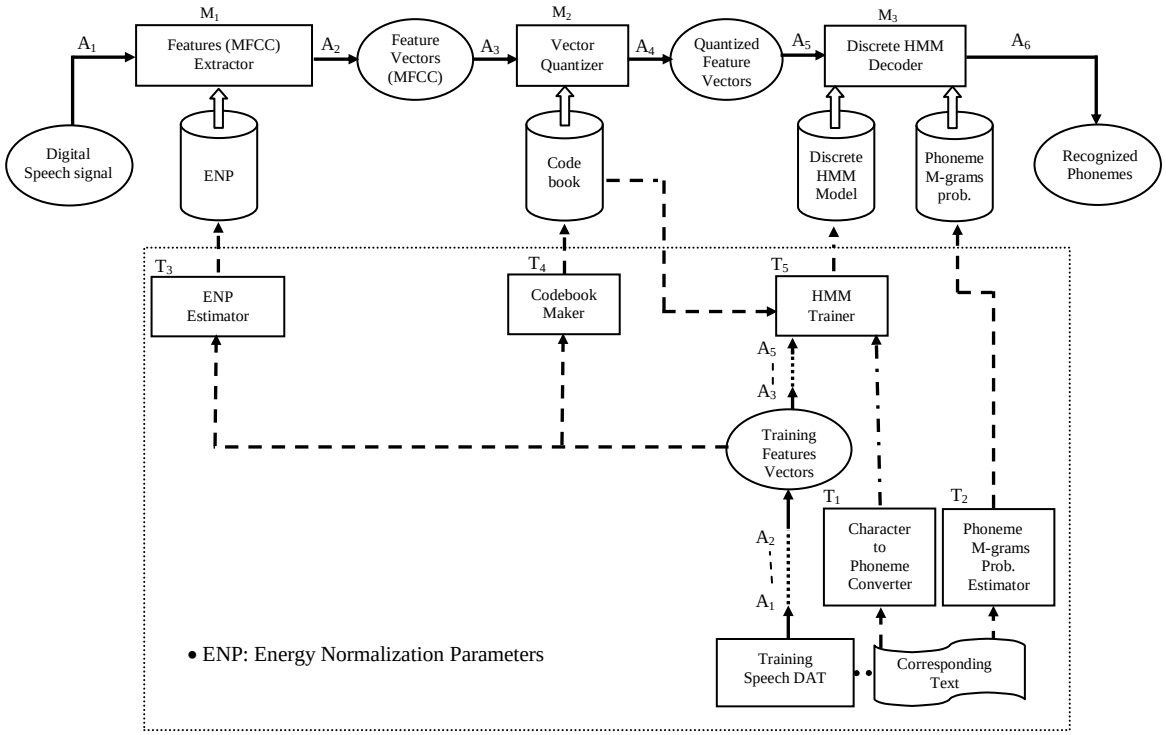

Fig. 8. Discrete HMM-based ASR system block diagram.

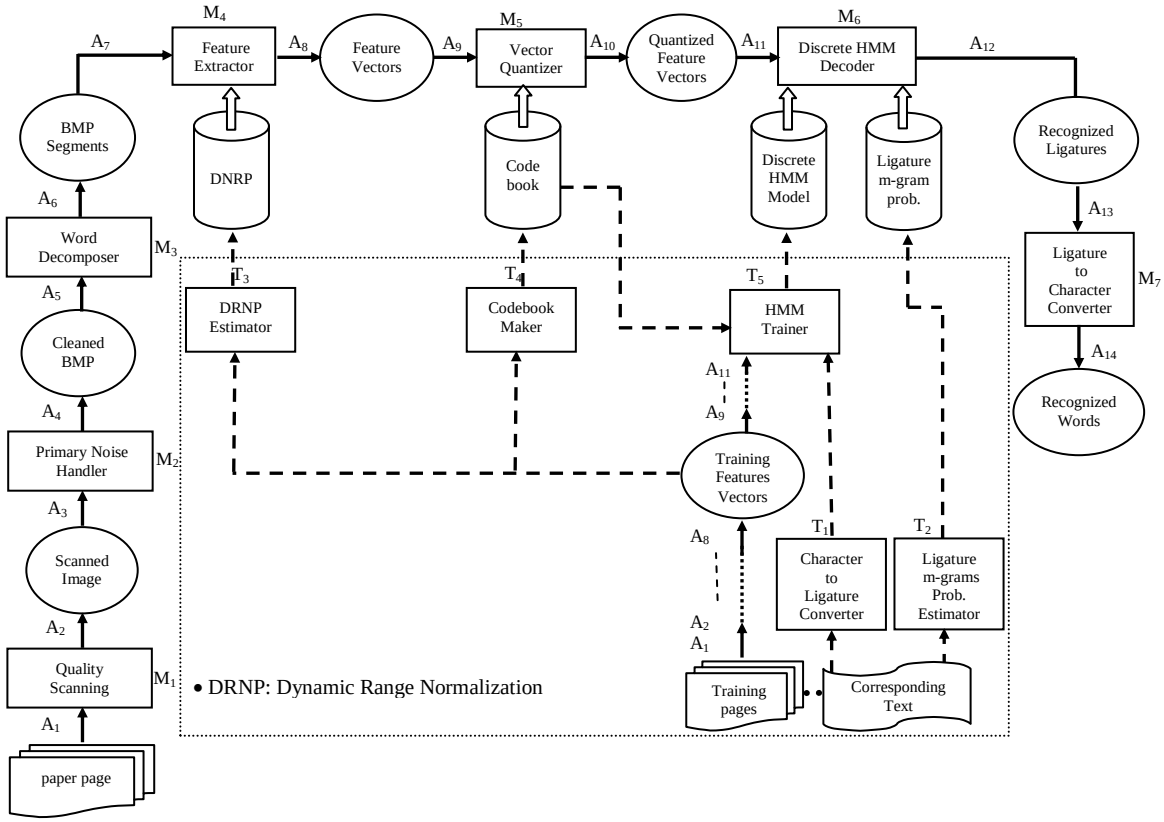

Fig. 9. Discrete HMM-based OCR system block diagram.

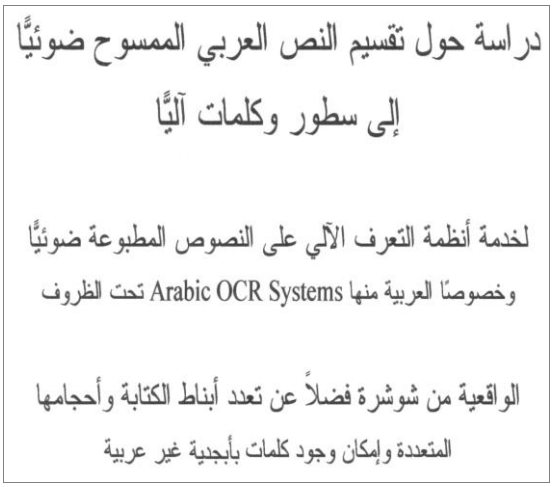

Fig. 10. Sample text rectangle bitmap before median filtering.

Fig. 11. Sample text rectangle bitmap after decomposition.

4.5 Vector Clustering and Quantization Modules

Taking input feature vectors, the vector-quantizer serially provides quantized observations to the discrete HMM decoder upon the recognition phase using a codebook generated by the codebook maker during the training phase. Using numerous sample text images of the training set of documents evenly representing the various Arabic fonts and sizes of interest, the codebook maker creates a codebook using the conventional Linde-Buzo-Gray (LBG) vector clustering algorithm [19] chosen for its simplicity and relative insensitivity to the randomly chosen initial centroids compared with the basic K-means algorithm [9]. The codebook size is a key factor that affects the recognition accuracy and is specified empirically to minimize the WER of the system.

4.6 Dynamic Range Normalization Parameters “DNRP” Estimator

This module is used to detect the effective dynamic range of the features to calculate the normalization parameters from the population of the feature vectors in the training data after pruning the extreme values. Normalization of the feature vectors injected to the vector quantizer balances the weights of the different dimensions of the features vector while calculating the distances between points in the features space during the recognition phase.

As the features population tends to follow a natural "Gaussian" distribution, so our estimation of the effective dynamic range is chosen to be within $\mu \pm 3\sigma$ of the distribution. The pruning step is essential before normalization, as the extreme values would render the resolution of the effective region of the features indiscriminative, which would in turn severely ruin clustering, hence the training and the recognition.

4.7 Characters-to-Ligature Converter

This module takes the text files corresponding to the training data to produce the sequences of grapheme tokens of all the ligatures included in each word using the predefined ligatures-set of each used font.

This module is built using a deterministic finite state machine. An example of character-to-ligature conversion is shown in Fig. 12. The sequence of ligature tokens along with the corresponding sequence of quantized feature vectors "i.e. observations" of each word are then forwarded to the discrete HMM trainer.

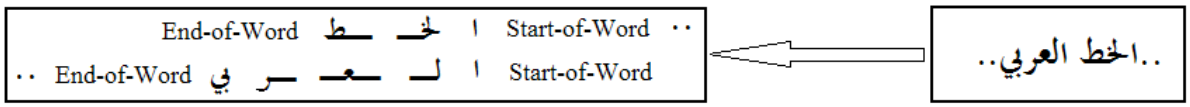

Fig. 12. Characters-to-ligatures conversion example.

4.8 Discrete HMM Trainer and Decoder

Our Arabic font-written OCR relies on HMM's with 1st order left-to-right topology [20] shown by Fig. 13. As one of the key factors that affect the system performance, the number of states per model is empirically fine tuned to optimize that performance.

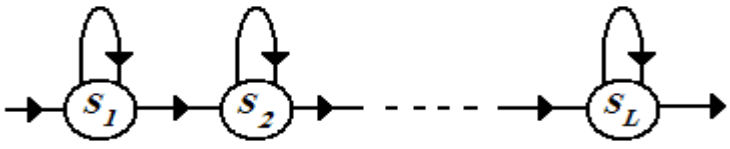

Fig.13. First-order left-to-right HMM 's topology.

Our discrete HMM trainer receives the observations sequence, which is the sequence of quantized feature vectors, of each word along with its sequence of ligature

tokens generated by the character-to ligature converter. Training the HMM 's are done over two steps: initialization and re-estimation embedded training [20, 21].

Initialization of HMM's is done using isolated ligatures where HMM parameters are initialized through two iterations; the first one uses Viterbi training as a hard deciding algorithm, while the second uses Baum-Welch re-estimation as a soft deciding algorithm.

Re-estimation embedded training which is the main HMM training step, uses the Baum-Welch re-estimation procedure to train the ligature models using the associated ligature tokens generated by the character-to-ligature converter to iteratively maximize the likelihood probabilities distributions $P(\underline{O}|\underline{I})$ [20].

The discrete HMM-based ligatures recognition problem is now rendered to obtaining the path $S_{f_g(t=1),f_s(t=1)},S_{f_g(t=2),f_s(t=2)},...,S_{f_g(t=N),f_s(t=N)}$ whose sequence of nodes accumulates the maximum sum of probabilities among all the possible paths over the search trellis shown in Fig. 14; where $N \geq \max_{i=1}^{M}(L_i)$ is the length of the sequence of observations. Any legitimate path can start at *t=1* only at the first state "marked by an inwards arrow" of any grapheme model, and can end only at *t=N* at the last state "marked by an outwards arrow" of any grapheme model.

As 1st order HMM 's are deployed; every state other than the last one in each HMM grapheme model $S_{i,k\leq L_i}$ represented by hollow nodes are connected at the next time instant only to either itself or its subsequent state in the same model by light links whose incremental probabilistic weights are respectively $\log(p(S_{i,k\leq L_i}|S_{i,k\leq L_i}))$ and $\log(p(S_{i,k+1\leq L_i}|S_{i,k\leq L_i}))$.

The conditional probability of a given observation O_t to occur at a given state $S_{i,k\leq L_i}$ is $\log(p(O_t|S_{i,k\leq L_i}))$. The 1st order approximation of the likelihood probability $p(\underline{O}|\underline{I})$ of the MAP criterion in Eq. (1) can then be expressed by the following formula

$$\log\left(p(\underline{I}|\underline{O})\right) \approx P_1^* = \sum_{j=1,f_s(j)\neq 1}^{j=N}\left(\log\left(p(S_{f_g(t=j),f_s(t=j)}|S_{f_g(t=j-1),f_s(t=j-1)})\right)\right)+\sum_{j=1}^{j=N}\left(\log\left(p(O_{j=t}|S_{f_g(t=j),f_s(t=j)})\right)\right) \quad (3)$$

The last state of each HMM grapheme model S_{i,L_i} represented by filled nodes are connected at the next instant either to itself, or to the first state in each HMM

grapheme model $S_{j,1}$ by dark links whose incremental probabilistic weights $\log(p(g_j \mid g_i))$ constitutes the grapheme-level bi-gram approximation of the language model in the MAP criterion (as shown by Eq. (1) above) $p(\underline{I})$ expressed as follows**,**

$$\log\left(p(\underline{I})\right) \approx P_2^* = \sum_{j=1,\, f_s(j)=1}^{j=N} \log\left(p(g_{f_g(t=j)} \mid g_{f_g(t=j-1)})\right) \tag{4}$$

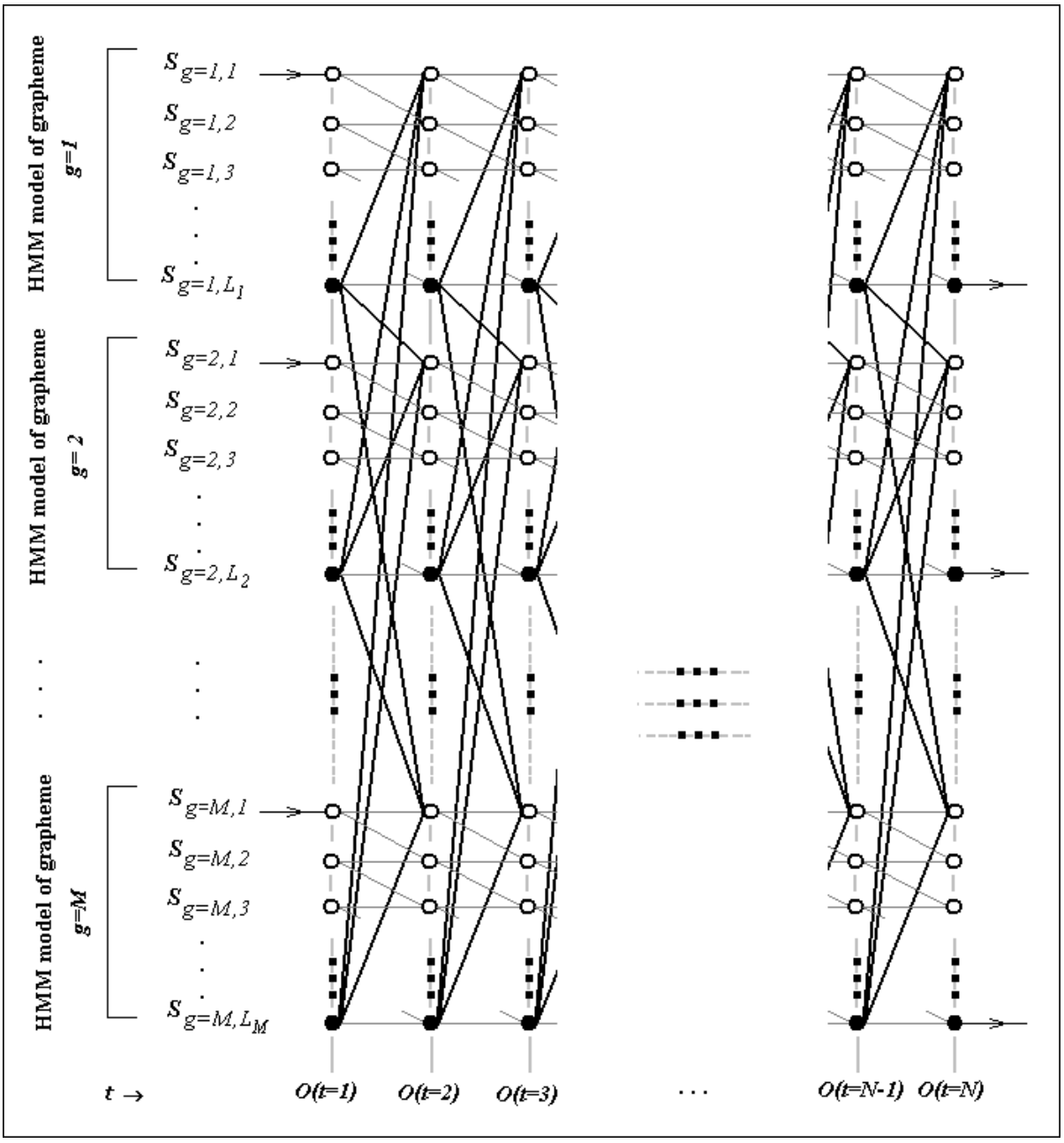

Fig. 14. HMM search trellis for graphemes recognition.

Given a sequence of observations, one can use the Viterbi algorithm to search though the above trellis to obtain the two functions f_g, f_s describing the optimal path whose states sequence correspond to the maximum $P_1^* + P_2^*$ all over the legitimately possible paths in the whole trellis.

4.9 Ligature m-Grams Probability Estimator

The Statistical Language Model "SLM" relies on the ligature frequencies in the text corpus to empirically construct the discrete probability distributions of grapheme "ligatures" sequences.

For recognition tasks; the SLM gives the a priori probability of a given hypothesis of ligature sequences. In our system, the bigram approximation of the ligatures SLM given by Eq. (5), is selected to match the 1st order Left-to-Right HMM topology chosen for our system.

$$p(g_j \mid g_i) = p\left(c_{j-m_2+1}^{j} \middle| c_{j-m_1-m_2+1}^{j-m_2}\right) \tag{5}$$

where m_1 & m_2 are the number of characters in the ligatures g_i and g_j respectively.

Arabic ligatures may be a compound of one, two, or three characters. By rewriting and expanding Eq. (5) using the chain rule for $m_1, m_2 \in \{1, 2, 3\}$, the possibilities of character bigrams, trigrams, …, 6-grams arise as follows in Eq. (6) below where the shorthand m-grams notation $p\left(c_{j-m}^{j}\right) \equiv p(c_{j-m}, c_{j-m+1}, \ldots, c_j)$ is used [15].

$$\begin{aligned}
\log\left(p(g_j \mid g_i)\right) &= && (m_1, m_2) = \\
&\log\left(p(c_j|c_{j-1})\right) && (1,1) \\
&\log\left(p(c_j|c_{j-2}^{j-1})\right) && (1,2) \\
&\log\left(p(c_j|c_{j-3}^{j-1})\right) && (1,3) \\
\log\left(p(c_{j-1}^{j}|c_{j-2})\right) &= \log\left(p(c_{j-1}|c_{j-2})\right) + \log\left(p(c_j|c_{j-2}^{j-1})\right) && (2,1) \\
\log\left(p(c_{j-1}^{j}|c_{j-3}^{j-2})\right) &= \log\left(p(c_{j-1}|c_{j-3}^{j-2})\right) + \log\left(p(c_j|c_{j-3}^{j-1})\right) && (2,2) \\
\log\left(p(c_{j-1}^{j}|c_{j-4}^{j-2})\right) &= \log\left(p(c_{j-1}|c_{j-4}^{j-2})\right) + \log\left(p(c_j|c_{j-4}^{j-1})\right) && (2,3) \\
\log\left(p(c_{j-2}^{j}|c_{j-3})\right) &= \log\left(p(c_{j-2}|c_{j-3})\right) + \log\left(p(c_{j-1}|c_{j-3}^{j-2})\right) + \log\left(p(c_j|c_{j-3}^{j-1})\right) && (3,1) \\
\log\left(p(c_{j-2}^{j}|c_{j-4}^{j-3})\right) &= \log\left(p(c_{j-2}|c_{j-4}^{j-3})\right) + \log\left(p(c_{j-1}|c_{j-4}^{j-2})\right) + \log\left(p(c_j|c_{j-4}^{j-1})\right) && (3,2) \\
\log\left(p(c_{j-2}^{j}|c_{j-5}^{j-3})\right) &= \log\left(p(c_{j-2}|c_{j-5}^{j-3})\right) + \log\left(p(c_{j-1}|c_{j-5}^{j-2})\right) + \log\left(p(c_j|c_{j-5}^{j-1})\right) && (3,3)
\end{aligned} \tag{6}$$

The probabilities in Eq. (6) are estimated using Bayes's_Good-Turing_Back-off methodology [15, 22]. The Arabic text corpus of the written language resource of the NEMLAR project [23] is used for building the SLM for our Arabic OCR.

4.10 Ligature –to- Character Converter

This module converts the sequence of recognized ligatures back into their equivalent plain-text character codes "e.g. ASCII codes".

5. EVALUATION EXPERIMENTS

Our system has been evaluated under the following experimental conditions:

- Training and testing pages are chosen arbitrarily from real-life documents which are full of punctuations and special symbols embedded within the text.
- The training database has been digitally acquired via a common but standard hardware of HP3800 flatbed scanner with TMA, and HP LASER jet 1100 printer.
- This database covers the 3 fonts stated in Table 1 below, each with 6 different sizes within the range of interest from 10 to 22 which is most common for real life uses.
- The sizes of the ligature sets vary from one font to another as shown in Table 1.
- Before the training, the alignment of the textual transcription files of the trained images versus the output of the lines & words decomposer is verified.
- The WER is measured following the speech recognition conventions; i.e. the number of substitutions, deletion, and insertions are summed up and divided by the total number of words in the input textual transcription files.

Table 1. The sizes of the ligature sets of the 3 fonts covered by the experiments.

	Simplified	Mudir	Traditional
Total # of ligatures	151	151	220
# of compound ligatures (> one character per	9	9	78
# of non alphabetic ligatures	25	25	25

6. RESULTS

Our experimental Arabic corpus consists of 540 pages of text scanned at 600 dpi, each associated with its textual transcription indicating the sequence of characters

in each word on each line. The underlying graphemes set of our transcription contains 220 ligatures including delimiters, numerals …, etc. This corpus represents 3 different fonts at 6 different sizes. Initialization is done using other 18 pages with each containing the isolated ligatures of each font at the specified size.

The number of HMM states chosen empirically to obtain the best WER is found to be 14 states per ligature, except for the few extremely wide ligatures where 18 states are used, and except for other few extremely slim ones where 7 states only are used .

Also, the codebook size is chosen empirically to be 1024 to cope with the large variance of all fonts' shapes, sizes, and styles, in order to obtain the best performance.

Training and testing the system have been done under two conditions:

i. Font-dependent; with the data of the same font at its different sizes.

ii. Multi-font, with the data of all the 3 fonts at their different sizes.

Table 2. shows the results under both conditions where about 80 % of the scanned lines are used for training and the rest are used as test data.

The lower WER with the Mudir font than the other two fonts might be attributed to its thicker contours and its more limited ligatures set the other two.

Table 2. Word error rate "WER" and character error rate "CER" of our OCR.

	Mono-font model		Multi-font model	
	WER	CER	WER	CER
Simplified	1.35 %	0.28 %	4.40 %	0.91 %
Mudir	0.84 %	0.18 %	1.00 %	0.21 %
Traditional	1.35 %	0.28 %	4.40 %	0.91 %
Average	1.18 %	0.25 %	3.30 %	0.69 %

7. CONCLUSIONS & FUTURE WORK

A full fledged Arabic font-written OCR system analogous to HMM-based ASR systems has been introduced and analysed. The system showed superior experimental performance over the similar reported systems when tried on wide sizes range of three of the most frequently used Arabic fonts under MS-Windows™, which is very promising in this field.

We are currently experimenting with this system over a much wider variety of Arabic fonts under both MS-Windows™ and Mac OS ™ environments. Moreover, generalization experiments are scheduled.

REFERENCES

1. Al-Badr, B., Mahmoud, S.A., "Survey and Bibliography of Arabic Optical Text Recognition", Elsevier Science, Signal Processing 41, pp. 49-77, 1995.
2. Govindan, V.K., Shivaprasad, A.P., "Character recognition; A review", Elsevier Science, Pattern Recognition, Vol. 23, No. 7, pp. 671-683, 1990.
3. Mantas, J., "An overview of character recognition methodologies", Elsevier Science, Pattern Recognition, Vol. 19, No. 6, pp. 425-430, 1986.
4. Gouda, A., "Arabic Handwritten Connected Character Recognition", PhD thesis, Dept. of Electronics & Electrical Communications, Faculty of Engineering, Cairo University, Nov. 2004.
5. Callan, J., Kantor, P., Grossman, D., "Information Retrieval and OCR: From Converting Content to Grasping Meaning", SIGIR conference, 2003.
6. Feng, S., Manmatha, R., "A Hierarchical, HMM-based Automatic Evaluation of OCR Accuracy for a Digital Library of Books", JCDL'06, June 11th–15th, 2006.
7. Bazzi, I., Schwartz, R., Makhoul, J., "An Omnifont Open-Vocabulary OCR System for English and Arabic", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 21, No. 6, June 1999.
8. Attia, M., "Arabic Orthography vs. Arabic OCR; Rich Heritage Challenging A Much Needed Technology", Multilingual Computing & Technology magazine, USA, Dec. 2004.
9. Duda, R.O., Hart, P. E., Stork, D. G., "Pattern Classification", 2nd ed., John Wiley & Sons Inc., 2001.
10. Davis, S.B., Mermelstein, P., "Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 28, pp. 357-366, 1980.
11. Bunke, H., Roth, T.M., Schukat-Talamazzini, E.G., "Offline Cursive Handwriting Recognition Using Hidden Markov Models", Elsevier Science, Pattern Recognition, Vol. 28, No. 9, pp. 1399-1413, 1995.
12. Attia, M., El-Mahallawy, M., "Histogram-Based Lines & Words Decomposition for Arabic Omni Font-Written OCR Systems; Enhancements and Evaluation", Lecture Notes on Computer Science (LNCS): Computer Analysis of Images and Patterns, Springer-Verlag Berlin Heidelberg www.SpringerOnline.com, LNCS 4673, 2007. Presentation is freely downloadable at http://www.RDI-eg.com/RDI/Technologies/paper.htm.
13. Gonzalez, R., Woods, R., "Digital Image Processing", 2nd ed., Prentice Hall, 2002.
14. Pratt, W.K., "Digital Image Processing", 2nd ed., John Wiley & Sons Inc., 1991.

15. Attia, M., Rashwan, M., Khallaaf, G., "On Stochastic Models, Statistical Disambiguation, and Applications on Arabic NLP Problems", The Proceedings of the 3rd Conference on Language Engineering; CLE'2002, the Egyptian Society of Language Engineering (ESLE). This paper is freely downloadable at http://www.RDI-eg.com/RDI/Technologies/paper.htm.
16. http://HTK.eng.cam.ac.uk.
17. Trier, O.D., Jain, A.K., Taxt, T., "Feature Extraction Methods For Character Recognition - A Survey", Elsevier Science, Pattern Recognition, Vol. 29, No. 4, pp. 641-662, 1996.
18. Khorsheed, M.S. "Offline Recognition of Omnifont Arabic Text Using the HMM ToolKit (HTK)", Pattern Recognition Letters-28, pp. 1563–1571, 2007.
19. Linde, Y., Buzo, A., Gray, R.M., "An algorithm for Vector Quantization Design", IEEE Transactions on Communications COM-28, pp. 84-95, Jan 1980.
20. Rabiner, L. R., Levinson, S.E., "A Speaker-Independent, Syntax-Directed, Connected Word Recognition System Based On Hidden Markov Model and Level Building", IEEE Transactions on Audio, Speech and Signal Processing, Vol. 33, No. 3, June 1985.
21. Young, S., et al., "HTK Book", Cambridge University Engineering, 2006
22. Katz, S.M., "Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer", IEEE Transactions on Acoustics, Speech and Signal Processing, vol. Vol. 35 no. 3, March 1987
23. http://www.NEMLAR.org/NEMLAR_resources.html.

عرض بنية ونتائج تقويم نظام للتعرف الضوئي آليًّا على الكتابة العربية
مناظر لأنظمة التعرف الصوتي على الكلام القائمة على نماذج ماركوف المخبأة

تم فى البحث الذى نحن بصدده تنقيح آليات أساسية توظف عادة فى أنظمة التعرف الضوئى آليا على الحروف مثل التقطيع المبدئى للسطور والكلمات واستخلاص الخصائص المميزة وغيرها، وذلك لبناء نظام للتعرف الآلى على الكتابة العربية ذى اعتمادية مناسبة لتطبيقات تكنولوجيا المعلومات ويعرض البحث أسلوب بناء أنظمة التعرف الآلى على الكتابة العربية المطبوعة القائم على نماذج ماركوف المخبأة عمومًا وللنظام الذي بنيناه خصوصاً، ويقدم النتائج التجريبية التى تبين تفوق النظام على نظم شبيهة.