

Autonomously Normalized Horizontal Differentials as Features for HMM-Based Omni Font-Written OCR Systems for Cursively Scripted Languages

Mohamed Attia¹, Mohsen A. A. Rashwan^{1,2}, Mohamed S. M. El-Mahallawy³

¹The Engineering Company for the Development of Computer Systems; RDI, Egypt

²Dept. of Electronics & Electrical Communication, Faculty of Eng., Cairo Univ., Egypt

{m_Atteya, Mohsen_Rashwan}@RDI-eg.com

³Arab Academy for Science & Technology and Maritime Transport; AAST-MT, Egypt

Mahallawy@AAST.edu

Abstract—Automatic font-written Optical Character Recognition (OCR) is highly desirable for numerous modern information technology (IT) applications. Reliable font-written OCR's for Latin scripts are readily in use since long. For cursively scripted languages, that are the mother tongues of over one fourth of the world population, such OCR's are however not available at a robust and reliable performance. In this regard, the main challenge is the mandatory connectivity of characters/ligatures (i.e. graphemes) that has to be resolved simultaneously upon the recognition of these graphemes.

Among the various approaches tried over decades, Hidden Markov Models (HMM)-based OCR's seem to be the most promising as they capitalize on the ability of HMM decoders to achieve segmentation and recognition simultaneously similar to the widely used HMM-based automatic speech recognition (ASR). Unlike ASR's, what is missing in HMM-based OCR's is the definition of a rigorously founded features vector capable to robustly achieving minimal “font type/size-independent” (omni-font) word error rates comparable to those realized with Latin scripts. Here comes the contribution of this paper that introduces such a sound features vector design, and experimentally shows its superiority in this regard.

I. INTRODUCTION

In order to be competent with the human capability at the digitization of printed text, font written OCR's should achieve an omnifont performance at an average word error rate (WER) $\leq 3\%$ and an average speed ≥ 60 words/min. per processing thread [1]. While font-written OCR systems working on Latin script can claim approaching such measures under favourable conditions, the best systems working on other scripts; especially cursive scripts like Arabic, are still well behind due to a multitude of complexities.

For example, the best reported ones among the few Arabic Omni font-written OCR systems can only claim assimilation WER's exceeding 10% under favourable conditions and not to mention the realistic ones [4], [10].

The mandatory connectivity of the “atomic graphical units of writing” (graphemes) is the major challenge in this regard, which calls for a solution that is simultaneously capable to realizing both of graphemes segmentation and recognition.

Inspired by the analogous problem of automatic continuous speech recognition typically approached via the hidden

Markov models (HMM) methodology [12], some promising attempts have been carried out to apply the same methodology for building HMM-based font-written OCR's on cursively scripted languages [4], [8], [10], [14]. While a remarkable success as unifont OCR's (like speaker-dependent ASR's) could be achieved, very few of them [4], [10] could only claim omnifont performance with low WER counted under favourable conditions (e.g. clean documents) of assimilation (not generalization) testing.

Our analysis of such systems revealed the weak point to be the lack of a sound features vector design adequate for the task of Omni font-written HMM-based OCR.

Graphemes connectivity, dotting, multiple graphemes for the same character at different positions, and composite ligatures are some of the extra challenging properties of Arabic script illustrated in figure 1 below [1], [3]. Being an extreme case of cursive scripts, we considered Arabic script as the first target to apply our new features vector design presented in section 3, and to evaluate the experimental results of this application in section 4 of this paper. For the convenience of the reader, the HMM-based OCR-to-ASR analogy is first reviewed in the next section.

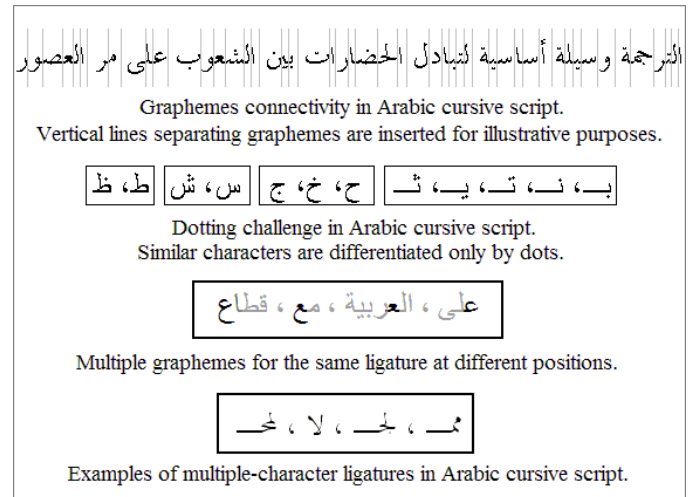


Fig. 1. Challenges of Arabic cursive script

II. ANALOGY TO HMM-BASED ASR

The system architecture of our HMM-based OCR schematized in figure 2 below may manifest the analogy with HMM-based ASR's. Like the digital acquisition of input speech signals, the target printed text page is digitized, cleaned, then the line as well as word boundaries are specified via the modules M_1 , M_2 , and M_3 respectively [2], [13].

For each word, like a speech utterance in ASR, signal features are sequentially observed through a narrow sliding window (i.e. frame) along the temporal/writing direction as illustrated by figure 3 on the next page. While the *Mel Frequency Cepstral Coefficients* (MFCCs) are considered as the well established standard features for digital speech processing [12], there is unfortunately no such effective nor agreed-upon counterpart for HMM-based OCR's [13], [14] !

After normalizing the dynamic range of each features-vector component, the module M_5 quantizes each vector versus a codebook inferred from a sample feature-vectors population during the training phase via the codebook making module T_4 using the famous LBG algorithm [11].

It is worthy to mention that vector quantization is not always a part of ASR's as most of them are continuous HMM-based ones like some HMM-based OCR's. However, our

OCR presented herein is based on discrete HMM due to the continuous/discrete hybrid nature of the components of its features vector whose design is detailed in section 3 of this paper.

Based on the maximum a posteriori probability (MAP) principle, the HMM decoder of module M_6 receives a sequence of quantized feature-vectors (i.e. observations) in order to infer the most likely sequence of corresponding graphemes according to the HMM models as well as the ligature m-gram probabilities constructed by the modules T_5 and T_2 respectively during the training phase [5], [8], [13].

In order to show the isolated impact of our new features vector design, the simplest 1st order L-to-R HMM topology as well as the standard Viterbi and Baum-Welsh algorithms are used for initialization, training, and recognition [12]. More elaborate HMM topologies and/or algorithms may be deployed later for further enhancements.

For cursively scripted languages like Arabic with fonts containing compound graphemes, the characters-to/from-ligatures mapping is achieved via the modules T_1 and M_7 respectively.

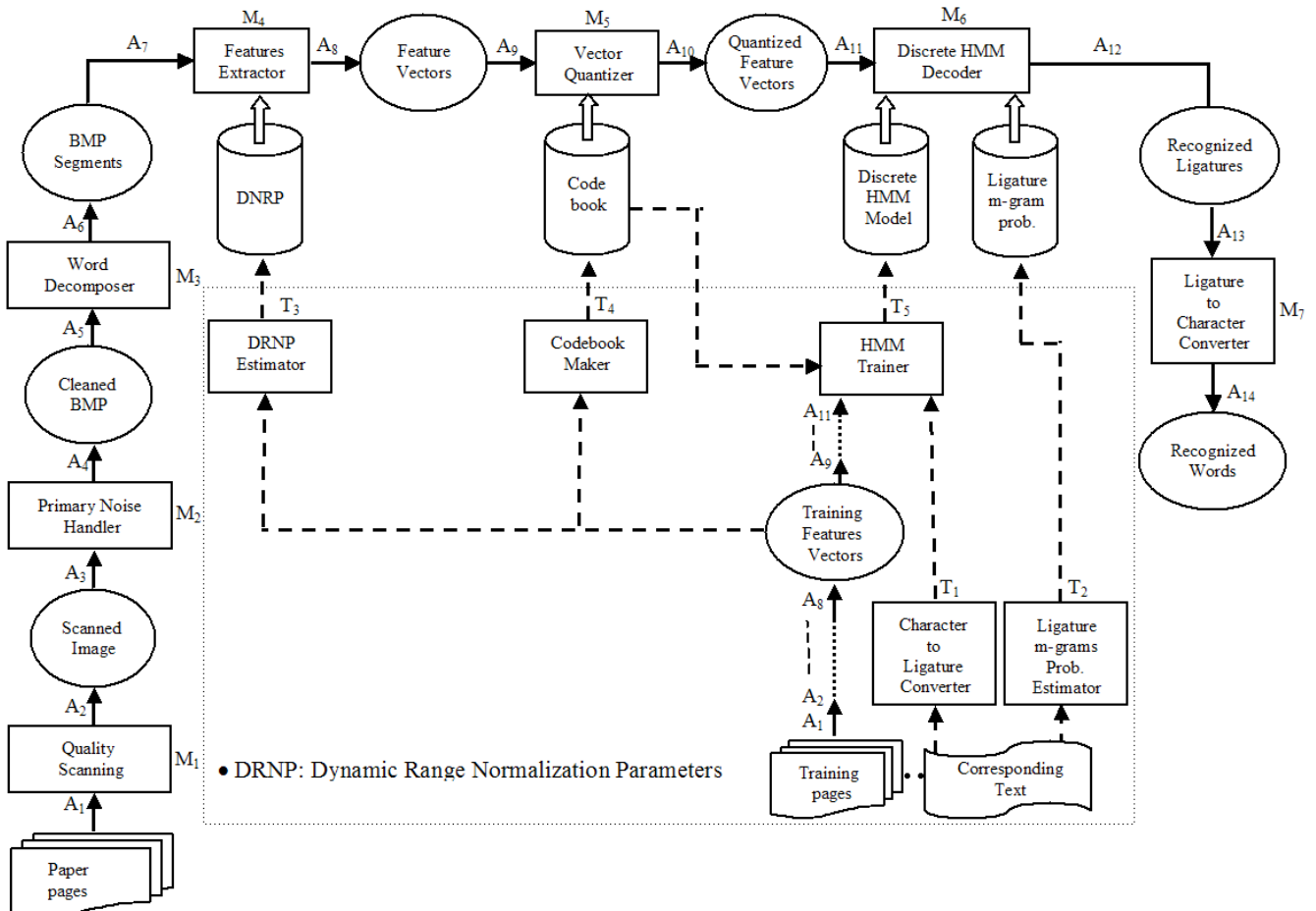


Fig. 2. System architecture of discrete HMM-based OCR analogous to HMM-based ASR.

A wide variety of features-vector designs have been tried through the published attempts for building Omni font-written HMM-based OCR's. Inspired by ASR's, some authors tried frequency domain oriented features which could only realize successful unifont (but never omnifont) recognition with cursive scripts! Despite the aforementioned analogy between the two problems, there are subtle differences regarding the content of the sliding window on both signals as shown by fig. 3 below.

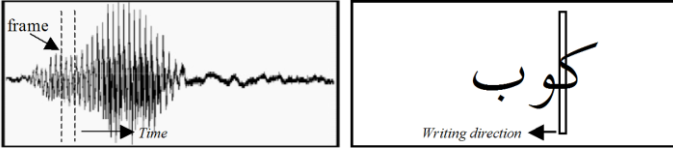


Fig. 3. Sliding window over a speech signal, and over an Arabic font-written text image bitmap

While the MFCCs of the signal content, within the speech frame of a typical width from 5 to 10 milliseconds, can somewhat time-invariantly reflect the local properties of the quasi stationary speech signal [12]-[14], frequency domain oriented features of the contents within a digitized text image frame of a typical width from one to a handful pixels would be crisply fluctuating due to the sharp luminosity edges from the writing contours to the background and vice versa. Such features with ASR can hence produce speaker-independent (i.e. Omni speaker) systems, while those features could result at best in unifont (i.e. speaker-dependent) OCR's due to the highly variant perturbations that can hardly capture generalized observation patterns corresponding to grapheme shapes.

Some other researchers tried to use the well known image characterizer of the seven invariant moments [8] which was also unable to realize omnifont recognition because these moments are integral in the sense that they are capable to distinguishing whole shapes regardless of their scale, tilt, and skew. Such an "Omni" capability vanishes when they are given differential slots on the graphemes.

Using simple time domain features [15], a third class of research groups realized some success towards an omnifont performance for cursive scripts; especially the Arabic one [4], [10]. Such features are mainly ad hoc combinations of simple partial measurements over the contents of the sliding window like its center of gravity, the number of black/white segments, the extensions of these segments, their relative positions within the same segment and maybe across the adjacent segments ... etc.

If such ad hoc time domain features partially describing the differential progress of cursively written words are able to realize limited omnifont performance, wouldn't autonomously normalized features that are neatly derived to fully and losslessly describe these differentials be capable to achieving a much superior performance?

III. THE NEW FEATURES VECTOR DESIGN

Within the HMM-based recognition methodology, the information observed through the signal content of the thin sliding window is differential by definition, which may hence

drive us intuitively to think of a differential design for the features vector.

Pursuing this line of thought; let each font-written word be formulated in the x - y plane as a parametrically represented contour $C(t)$; $x=g_x(t)$, $y=g_y(t)$.

Regarded as an analog signal, the width of the sliding window would infinitesimally shrink into the differential dx , and the simplest features vector to be thought of is the total derivative of the contour along the horizontal writing direction so that...

$$\underline{f}(x) = \frac{d_x \underline{C}(x, y)}{dx} \quad (1)$$

Such a features vector would be fully representative to the writing contours which are losslessly retrievable via integration. Turning our sight back to the actual digitized form of the writing contours; $\underline{C}^*(x, y)$, the differential is then best approximated by a width of a single pixel and our features vector turns into the total difference of the digitized contour along the horizontal writing direction so that..

$$\underline{F}_i = \Delta_x \underline{C}^*(x, y) \Big|_{x=i} \quad (2)$$

In the absence of closed-form expressions of the writing contours, how can this total difference be practically encoded into a manageable features vector? This is the non-trivial question that arises at this point. Any viable answer must satisfy the following requirements:

- 1) That vector must have a finite dimensionality.
- 2) This dimensionality must also be fixed for any vector extracted from the contents of the sliding window at any $x=i$.
- 3) This dimensionality should be kept as compact as possible.
- 4) All the components of that vector should be computable efficiently.
- 5) Sequences of feature vectors corresponding to the same ligatures sequence should remain maximally invariant versus the irregularities irrelevant to the writing concepts in the target script; e.g. font and scale.

While failing to satisfy the first two requirements shifts such vector designs completely out of the HMM recognition paradigm, the failure at satisfying the next couple would produce an unacceptably expensive implementation as per the time and/or storage at both the training and recognition phases.

Meeting the fifth requirement is a primal factor towards realizing an omnifont (i.e. font type/size-independent) performance of the sought OCR.

One key idea towards producing a features vector design satisfying such a criteria is to consider that each single-pixel window (henceforth called a slice) at $x=i$ contains a limited maximum number of vertical dark segments; symbolized by M . For example, setting $M=4$ for Arabic script is appropriately sufficient.

The other key idea is to differentially express each segment; $S_{j,i}$ in a sub features vector; $\underline{S}_{j,i}$ composed of definitely 4 components encoding all the agglomerative and topological features relative to the segments in the preceding slice at $x=i-1$. With a dimensionality of $4 \cdot M$, the features vector \underline{F}_i derived from each slice at $x=i$ is then constructed by the union of its sub feature vectors as follows:

$$\underline{F}_i = [\underline{S}_1|_{x=i} \cup \underline{S}_2|_{x=i} \cup \dots \cup \underline{S}_M|_{x=i}] = \bigcup_{j=1}^M \underline{S}_{j,i} \quad (3)$$

To explain how to compute the abovementioned sub feature vectors, consider figure 4 below illustrating a hypothetical pixel-level view of two consecutive slices:

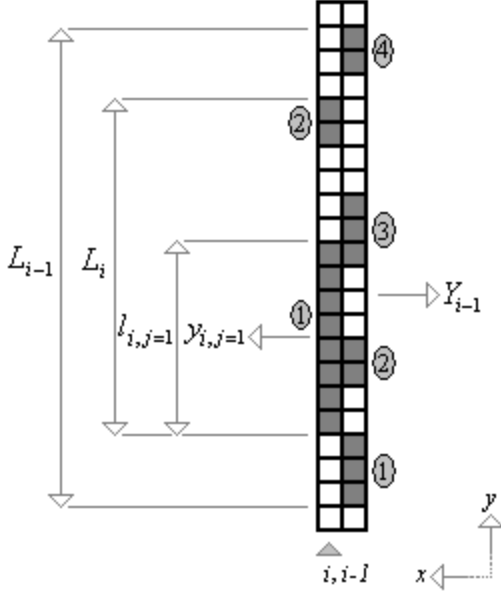


Fig. 4. Pixel-level view of two consecutive slices

Where the segments of each slice are ordered in a bottom-up manner, $l_{i,j}$ & $y_{i,j}$ are respectively the height and center of gravity of segment $S_{j,i}$, and L_i & Y_i are respectively the height and center of gravity of the whole slice.

A segment is considered disconnected if it is in 8-connection with none of the segments in the preceding slice. A whole slice is considered disconnected, if its preceding slice is empty.

As all the feature vectors must have definitely $4 \cdot M$ dimensions, the feature vectors derived from slices containing less than M dark segments are padded with nullified sub feature vectors that are assumed to correspond to empty segments.

For any given segment, only 5 distinct topological possibilities can exhaustively be considered. For each possibility, one formula from those numbered 4 to 8 below is applied to compute the sub feature vectors whose first two components encode the agglomerative properties while the last two encode the topological properties of segments relative to the preceding slice. Here are the 5 topological possibilities along with the formulae to compute the sub feature vectors at each:

1) For non-empty segments in the initializing slice of the whole word:

$$\underline{S}_{j,i=1} = \left[\frac{y_{1,j} - y_{1,1}}{L_{1,1}}, \frac{l_{1,j}}{L_{1,1}}, -1, -1 \right] \quad (4)$$

2) For non-empty disconnected segments in connected slices:

$$\underline{S}_{j,i} = \left[\frac{y_{i,j} - Y_{i-1}}{L_{i-1}}, \frac{l_{i,j}}{L_{i-1}}, -2, -2 \right] \quad (5)$$

3) For non-empty segments in disconnected slices other than the initializing one of the word:

$$\underline{S}_{j,i} = \left[\frac{y_{i,j} - Y_{i-k}}{L_{i-k}}, \frac{l_{i,j}}{L_{i-k}}, -3, -3 \right] \quad (6)$$

Where the slice at $x=i-k$ is the first non-empty slice before the one at $x=i$.

4) For all empty (i.e. null) segments:

$$\underline{S}_{j,i} = [0,0,0,0] \quad (7)$$

5) Otherwise:

$$\underline{S}_{j,i} = \left[\frac{y_{i,j} - Y_{i-1}}{L_{i-1}}, \frac{l_{i,j}}{L_{i-1}}, s_{START,i-1}, s_{END,i-1} \right] \quad (8)$$

Where the last two components of formula no. 8 respectively determine the orders of the most bottom and top segments in the previous slice $x=i-1$ at which are in 8-connection with $S_{j,i}$.

For example, the features vector at $x=i$ in figure 4 above can be simply calculated using formulae no. 8, 5, and 7 respectively as follows:

$$\begin{aligned} \underline{F}_i &= \left[\left[\frac{-1.9}{20}, \frac{8}{20}, 1, 3 \right] \cup \left[\frac{7.1}{20}, \frac{2}{20}, -2, -2 \right] \cup [0,0,0,0] \cup [0,0,0,0] \right] \\ &= [-0.095, 0.4, 1, 3, 0.355, 0.1, -2, -2, 0, 0, 0, 0, 0, 0] \end{aligned}$$

IV. EXPERIMENTAL RESULTS AND EVALUATION [7]

The training data, used to experimentally evaluate the performance of the introduced features vector within the presented HMM-based font-written OCR, covers 9 visually distinct and widely used Arabic fonts under both the MS-Windows and Macintosh OS environments. For each font, 6 different sizes within the typical document-editing range (10 to 24; say) are represented by 25 pages per size. Each page contains about 200 words selected spontaneously from contemporary Arabic websites. Assuming an average of 4 ligatures per Arabic word [3], [13], the size of the training data $\approx 9 \times 6 \times 25 \times 200 \times 4 = 1.08M$ graphemes. All the training and testing pages are digitized at B&W 600 dpi then losslessly stored as bitmaps. A line-boundaries aligned text file (called the label file) corresponding to each of these

bitmaps is also supplied in the labelled training data. A table of the ligatures set defined per each font in the training data is also provided.

The testing data is composed of two parts; assimilation testing data, and generalization testing data:

Assimilation testing data represents all the above mentioned fonts and sizes using 5 pages (other than the 25 pages of training) per size per font. The size of assimilation testing data $\approx 9 \times 6 \times 5 \times 200 \times 4 = 216K$ graphemes. Text label files are also provided.

Generalization testing data covers 3 Macintosh OS fonts other than the 9 fonts used for training. For each of these 3 fonts, 6 different sizes within the typical document-editing range are represented by 5 pages per size. The size of generalization testing data $\approx 3 \times 6 \times 5 \times 200 \times 4 = 72K$ graphemes. Text label files are also provided.

In order to cover all the significant regular Arabic fonts, 220 ligatures have been defined in our system which is a large number of distinct patterns compared with few tens typically needed to deal with Latin scripts.

The parameters setting of our HMM-based OCR during both the training and recognition phases are summed up in table I below.

TABLE I
HMM PARAMETERS SETTING

HMM type	Discrete, 1 st order, L-to-R
Codebook size	2,048 Centroids
No. of HMM states per model	7 States for very slim graphemes. 18 States for very wide graphemes. 14 States for normal graphemes.
HMM algorithms	<u>Initialization</u> : Viterbi and Baum Welsh <u>Embedded training</u> : Baum Welsh <u>Recognition</u> : Viterbi
Training data size	1,080,000 Graphemes
Assimilation testing data size	216,000 Graphemes
Generalization testing data size	72,000 Graphemes

The bigram ligature-level statistical language model deployed in our HMM decoder is built from the domain balanced corpus of the NEMLAR project whose size $\approx 550K$ words $\approx 2.2M$ graphemes [16]. A Bayesian probability estimator refined by the Good-Turing discount and Back-Off techniques [9] is used for this purpose.

Under these conditions; assimilation and generalization testing have been conducted to obtain the overall recognition error rates. To study the effect of the statistical language model (SLM), both experiments are run twice; with the SLM enabled on the first the run, and with the SLM neutralized on the second run.

The obtained error rates of the 4 cases are summed up in table II below where the assimilation error rates are designated by the subscript A, and the generalization error rates are designated by the subscript G.

TABLE III
EXPERIMENTAL RESULTS

SLM enabled	CER_A	0.77%
	WER_A	3.08%
	CER_G	2.58%
	WER_G	10.32%
SLM neutralized	CER_A	1.53%
	WER_A	6.13%
	CER_G	3.65%
	WER_G	14.60%

WER is the word error rate which is the error rate perceived by the users of the OCR, and is hence the useful error rate. On the other hand, CER is the character (or more accurately; the ligature) error rate which is the one typically reported in the scientific literature as well as the commercials. It can easily be shown that $WER \approx \ell \cdot CER$; $CER \ll 1$, and ℓ is the average no. of ligatures per word which ≈ 4 for Arabic.

The error rates obtained after the generalization test with the SLM enabled are the most indicative as these conditions are closest to the real life ones. Given that both the scientific and commercial literature in this area typically report about $CER_A < CER_G$, the experimental results of table 3 above puts our HMM-based OCR with the new features vector in the lead among the other open-vocabulary Omni font-written OCR's operating on cursive scripts; at least for Arabic [1], [4], [8], [10], [15].

V. CONCLUSION

The modest status quo of font-written OCR of the challenging cursive scripts compared with its counterpart of Latin scripts has first been reviewed. The HMM-based approach, that proved to be the most promising for tackling this problem, has then been dissected to reveal its Achilles' heel of lacking a sound features vector design like the MFCCs for HMM-based ASR's.

The paper then proceeds to present its main contribution of rigorously deriving a new sound features vector design based on horizontal differentials as autonomously normalized lossless features for this kind of OCR.

Extensive assimilation and generalization testing has both been carried out over the Arabic script which is an extreme case of cursive scripts. These testing experiments have been described in detail, and the obtained results have finally been presented to show how this new design goes beyond the other ones tried so far with HMM-based OCR's operating on cursive scripts.

REFERENCES

- [1] B. Al-Badr, S. A. Mahmoud, *Survey and Bibliography of Arabic Optical Text Recognition*, Elsevier Science, Signal Processing 41, pp. 49-77, 1995.
- [2] M. Attia, M., El-Mahallawy, *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, LNCS 4673, 2007.
- [3] M. Attia, *Arabic Orthography vs. Arabic OCR*, Multilingual Computing & Technology magazine, USA, Dec. 2004.

- [4] I. Bazzi, R. Schwartz, J. Makhoul, *An Omnifont Open-Vocabulary OCR System for English and Arabic*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 21, No. 6, June 1999.
- [5] H. Bunke, T. M. Roth, E. G. Schukat-Talamazzini, *Offline Cursive Handwriting Recognition Using Hidden Markov Models*, Elsevier Science, Pattern Recognition, Vol. 28, No. 9, pp. 1399-1413, 1995.
- [6] R. O. Duda, P. E. Hart, D. G. Stork, *Pattern Classification*, 2nd ed., John Wiley & Sons Inc., 2001.
- [7] M. S. M. El-Mahallawy, *A Large Scale HMM-based Omni Front-Written OCR System for Cursive Scripts*, PhD thesis, Dept. of Computer Engineering, Faculty of Engineering, Cairo University, Apr. 2008; http://MEDAR.info/Archive/Papers_Articles_Theses.php.
- [8] A. M. Gouda, *Arabic Handwritten Connected Character Recognition*, PhD thesis, Dept. of Computer Engineering, Faculty of Engineering, Cairo University, Nov. 2004.
- [9] S. M. Katz, *Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer*, IEEE Transactions on Acoustics, Speech and Signal Processing, vol. ASSP-35 no. 3, March 1987.
- [10] M. S. Khorsheed, *Offline Recognition of Omnifont Arabic Text Using the HMM ToolKit (HTK)*, Pattern Recognition Letters-28 pp. 1563-1571, 2007.
- [11] Y. Linde, A. Buzo, R. M. Gray, *An algorithm for Vector Quantization Design*, IEEE Transactions on Communications COM-28, pp. 84-95, Jan 1980.
- [12] L. R. Rabiner, S. E. Levinson, *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, Volume 33, Number 3, June 1985.
- [13] M. Rashwan, M. W. T. Fakh, M. Attia, M. El-Mahallawy, *Arabic OCR System Analogous to HMM-Based ASR Systems; Implementation and Evaluation*, Journal of Engineering and Applied Science, Cairo University, www.Journal.eng.CU.edu.eg, Dec. 2007.
- [14] M. Rashwan, *A New OCR System Similar to ASR System*, The 10th International Conference on Computing and Information, ICCI 2000, Kuwait, Nov. 18th-21st, 2000.
- [15] O. D. Trier, A. K. Jain, T. Taxt, *Feature Extraction Methods For Character Recognition - A Survey*, Elsevier Science, Pattern Recognition, Vol. 29, No. 4, pp. 641-662, 1996.
- [16] M. Yaseen, M. Attia, B. Maegaard, ..., *Building Annotated Written and Spoken Arabic LR 's in NEMLAR Project*, LREC2006 conference <http://www.LREC-conf.org/LREC2006>, Genoa-Italy, May 2006.