Efficient Generation of Comprehensive Database for Online Arabic Script Recognition

Raid Saabni
Computer Science Department
Ben-Gurion University Of the Negev, Israel
Traingle R&D Center, Kafr Qarea, Israel
saabni@cs.bgu.ac.il

Jihad El-sana
Computer Science Department
Ben-Gurion University Of the Negev
Beer Sheva, Israel
el-sana@cs.bgu.ac.il

Abstract

The difficulties in segmenting cursive words into individual characters have shifted the focus of handwriting recognition research from segmentation-based approaches to segmentation-free (holistic) methods. However, maintaining and training large number of prototypes (models) that represent the words in the dictionary make the training process extremely expensive and difficult in computing resources. In this paper we present an efficient system that automatically generates prototypes for each word in a given dictionary using multiple appearance of each letter shape. Multiple appearance allows for many permutation of shapes for each word and thus complicates searching for the right prototype. To simplify the training, reduce the maintained prototypes, and avoid over fitting, we used dimensionality reduction followed by clustering techniques to reduce the size of these sets without affecting their ability to represent the wide variations of the handwriting styles. A set of generated fonts are created by professional writers imitating all handwriting styles for each character in each position. These Fonts are used to generate all shapes for writing each word-part in a comprehensive dictionary. Principal component analysis and k-means clustering techniques are performed to select the minimal number of shapes representing the wide variations of handwriting styles for a word-part. Experimental results using an online recognition system proves the credibility of this process compared to manually generated databases.

1. Introduction

The need for advanced human-computer interface such as online handwriting recognition systems drives research in handwriting script recognition. Handwriting is a compact solution in the sense it compromises efficiency with size to enable creating texts in limited devices, which makes it a natural choice for portable small devices. However, recognizing cursive handwriting is still difficult because of the huge variance and individuality of personal handwriting. Arabic script is naturally cursive both in handwriting and print. The segmentation based approach starts with segmenting words into individual characters, which are then recognized and combined to identify the written word. The segmentation free approach, from the other hand, processes a whole word to be classified at once without segmenting to characters. The segmentation based approach is prone to errors, which make the holistic approach the leading technique used in research in handwriting recognition. Unfortunately, this approach introduces the need for huge datasets for efficient and extensive training, especially for large vocabularies.

The cursiveness of Arabic scripts forced researchers to reduce the domain size by using small dictionaries or recognizing only isolated forms of letters. In recent years, as in other scripts, the holistic approach became a major stream in Arabic handwriting recognition research. This trend creates an urgent need for developing wide and complete databases for training and testing.

In this paper we present a new approach for efficient production of a synthetic comprehensive database. This database includes many shapes for each word while using and imitating different handwriting styles. In this system we use a novel approach to produce synthetic shapes of any Arabic word using predefined handwriting fonts representing all the possibilities of writing each letter in the different positions. To keep the database compact thereby reducing redundancy, we used clustering and dimensionality reduction techniques. Principal Component Analysis and clustering methods were used to select a compact representative samples. The aim is to cover the huge variety of writing styles of each word-part while keeping the size as minimal as possible to enable affordable computation.

The rest of this paper is organized as follows: in section 2

we give a short introduction to Arabic script characteristics; in section 3 we explore the closely related work to Arabic script recognition and databases; a full description of the proposed system stages is given in section 4. Section 5 and 6 presents the results and discuss directions for future work.

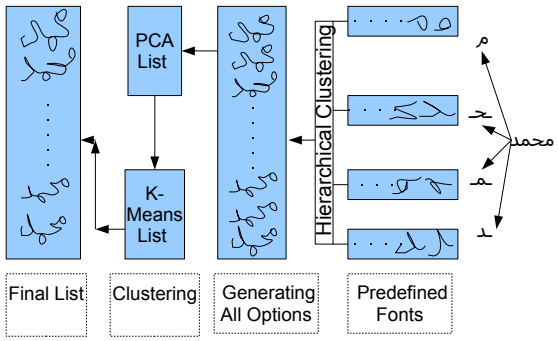

Figure 1. A diagram flow sample for generating a compact set of shapes for a given word-part.

2. Characteristics of Arabic script

The Arabic aleph bet is widely used for more than twenty different languages such as Farsi, Urdu, Malay, Hausa, and Ottoman Turkish. Arabic script is written from right to left in a semi-cursive manner. It is similar to western scripts, in that it has a strict alphabet consisting of letters, numerals, punctuation marks, spaces, and special marks. On other hand, it is different in the way it combines letters into words and the way it treats vowels. The Arabic script consists of 28 basic letters and 12 additional special letters. The shape of a letter in Arabic scripts usually changes due to the different position within the word – *initial, medial, final, and isolated* – according to its adjacent letters. As a result, the Arabic script has 120 different shapes representing all the letters in all positions within a word. Among the basic letters, six letters interrupt the cursiveness of a word by prohibiting a connection to the following letters which splits words into connected groups of letters called *word-parts*. In addition to the main body, many Arabic letters have a set of additional strokes that may include dots and short strokes. These additional strokes are mostly written below or above the letter body and are usually called delayed strokes as they are written after completing the main stroke of the component. Each component includes one or more letters and with its additional strokes forms a part of a word called *word-part*. Additional strokes *(unlike diactrics)* are mandatory in Arabic scripts and their absence may cause ambiguities. Their unique order and existence can be used to filter and eliminate candidates in classification processes. As an example, the word-parts (*kabeer* (كبير) and *katheer* (كثير)) are different word-parts with the same main body and different sets of additional strokes. Using one set of additional strokes eliminates one possibility and keeps the other.

We have explored a large collection of Arabic texts and extracted $300,000$ different words combined of $82,000$ different word-parts. Ignoring the additional strokes reduced the number of different word-parts to $40,000$. Since the text collection was very large and came from different domains and periods, we believe these results closely represent the Arabic language (see [15] for more details). In this research we use the dictionary of the $82,000$ word-parts to synthesize numerous shapes of each word-part while ignoring the additional strokes. These shapes are used to generate a complete database of shapes for the Arabic script.

3. Related work

The research of Arabic script recognition came to the attention of researchers very recently compared to Latin and Chinese. This delay exposed researchers to results and techniques from other scripts which were adapted and improved to be used for Arabic scripts. Segmenting cursive Arabic words to characters is a hard task due to the huge variety of different writing styles and the absence of constraints and consistency. Attempts have been made to recognize isolated forms of Arabic letters, avoiding the segmentation process, by obligating a non-cursive style of writing [8, 11, 4, 12]. Segmentation-based methods, where words are initially segmented to character and later recognized were developed or adapted and improved from other scripts [7, 6, 2, 9, 16]. As in other scripts, the poor recognition rates results from unsuccessful segmentation shifted the focus to the segmentation free approach. In the holistic approach [9, 10, 3, 14], complete words are processed to be recognized bypassing the character segmentation stage. Obviously, as opposed to the holistic approach, the other two options do not require large databases for training.

The holistic approach was initially used for small vocabulary tasks, such as check verification, mail sorting, and key word-searching. Lately a good deal of attention has been addressed towards seeking efficient methods targeting script recognition for large vocabularies using the holistic approach. Part of the challenges facing this approach are the large databases for training and testing as well as time and space efficiencies.

Many databases for handwritten script recognition tasks *(UNIPEN,CEDAR,NIST,IRONOFF,etc)* had been developed for English scripts. In contrast, very few databases were developed for the Arabic script and fewer became publicly

available. Among these, we can mention *IFN/ENIT and CEDAR* databases, but mostly researchers had developed their own small datasets or large databases that are not available to the public [13, 1, 5, 14]. None of these databases included all the different handwritten words or word-parts in the Arabic language. As a result, there is no standard comprehensive database (on-line or of-line) for Arabic handwriting script recognition.

4. Our Approach

The primary task of character recognition is to take an input image of a sequence of characters and correctly assign it to the possible output class, which should be a list of valid words hopefully including the correct word. The holistic approach avoids the obstacles of segmentation, but has to use a large number of shapes of the all different words in Arabic. Hence, large sets of samples are essential to enable efficient training and accurate classification. Such large sets become very expensive and time consuming when dealing with open vocabulary recognition systems. In this paper we present a novel approach to generate such large sets synthetically. Professionally trained writers created predefined sets of different handwriting styles (*Fonts*) using their own handwriting style. These sets of predefined fonts were used to generate all shapes and permutations for writing each word-part in the lexicon. This process results large sets of shapes, then Principal Component Analysis (PCA), Hierarchical and K-means clustering were used to reduce the size of these large sets by eliminating redundant shapes. The steps of the proposed system are: $1)$Generating Handwriting Fonts , $2)$Synthetic word-parts Generation and $3)$Clustering and Dimensionality Reduction. In the rest of this section we discuss each step in detail.

4.1. Generating Handwriting Fonts

We have developed a tool to enable writers to generate a set of predefined fonts by drawing different shapes for each letter form. The writers use their handwriting styles but also imitate different common writing styles for ligatures or letters in different positions. For example, writers were asked to write the pair letters (حم) in their own writing style, but also they were advised to write them vertically tiled in order to adjust to the vertical tiling style, (see Figure 2). Each letter in each position was written by the same writer several times. The average number of shapes written for a letter in a unique position was around 20 and up to 50 in some cases. The writers were guided to give special attention to special ligatures such as (لا) and other common pairs of consequent letters which may not concatenate horizontally. For this purpose, our system provides two schemes. In the first one, writers wrote one letter in one form each time while in the second they wrote a complete word-part which had been separated manually to individual characters. This stage provides vectors of two-dimensional points taken from the written strokes. To ignore redundant shapes of the same letter written by different writers, we perform hierarchical clustering. The result of this stage is a set of fonts which include different shapes writing each letter in all forms.

4.2. Synthesizing Word-part Shapes

Template and statistical matching techniques require large datasets of predefined templates to train the system. These sets could categorized into m-writer-dependent and writer-independent. The writer-dependent systems usually use a set of predefined templates derived from the handwriting of the trainers (writers). The writer-independent systems use sets of predefined large collection templates that aim to represent large variety of handwriting styles.

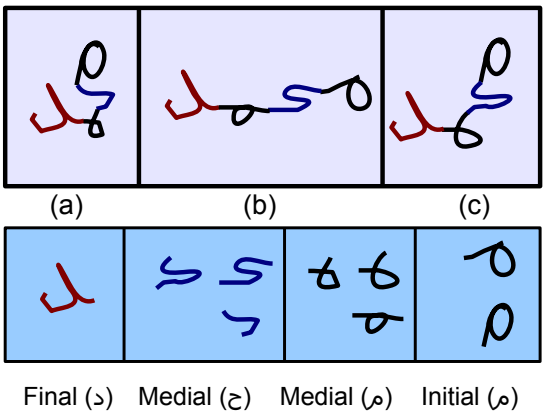

Figure 2. Three samples for synthetically generating the word "محمد".

In this research, we focus on generating writer-independent datasets, but the writer-dependent datasets could be easily constructed using the same approach. The set of predefined fonts generated in the previous step is used to generate a writer-independent open vocabulary database of word-parts shapes. For each word-part ω in the lexicon Σ (In our system, we have used an Arabic word-parts lexicon that includes almost every word-part in the Arabic language – around $82,000$ word-parts), the system determines

the shape of the letters in ω, while taking into account the position of each letter and ignoring the additional strokes. The shapes representing ω are generated by concatenating the various shapes for each letter in the right order, thus generating all the possible permutations of ω. The concatenation is simple and performed by stitching the endpoint of each letter shape to the start point of the following one and smoothing the stitching region.

Let $V_i(l,p)$ be a vector $(v_{i0}, \cdots, v_{in})$ representing the shape i of letter l in the position p. To generate one shape for the word-part $\omega = (l_1, l_2, \cdots, let_m)$ with length m, we concatenate the vectors $V_{i1}(l_1, Ini)$, $V_{i2}(l_2, Med)$,$\cdots$, and $V_{im}(let_m, Fin)$ each represents one appearance of the given letter and form, Where Ini, Med, and Fin stands for the $Initial$, $Medial$,and $Final$ positions of a letter within a word-part, respectively. The concatenation is done by joining the endpoint of the vector $V(l_i, p_i)$ with the start point of $V(l_{i+1}, p_{i+1})$. In the appending (Concatenation) process, points in the vector $V(Let_{i+1}, Pos_{i+1})$ are adjusted to be aligned to the previous vector using the Euclidean distance between its start point and the end point of the previous vector. The resulting vector contains all the points of one possible shape of the vector $V(l_i, p_i)$ for each letter l_i within the word-part ω after aligning in the concatenation process as explained previously(see Figure 2).

This process is done for each combination of all the different shapes representing each letter within the word-part ω in the predefined fonts. Obviously, no appending process is done when the length of the word-part is one and the concatenation of the initial and medial vectors is done when the length is two. As expected, the results of this step are huge sets of shapes for each word-part in the lexicon. For example, with a word-part that contains five letters with eight different shapes for each letter the method produces a list of $38880 = 5^7$ different shapes. It is obvious that many shapes are redundant as they display only minor differences, thus calling for techniques to reduce redundancy.

4.3. Clustering and Dimension Reduction

The generated representation for each word-part in the lexicon is too large for practical use. Fortunately, we have realized that large fraction of these representations have very few or no differences with high percentage of redundancy, which could be reduced dramatically by using clustering and dimension reduction techniques. Those large sets of shapes generated for each word-part in the lexicon, may include tens of thousand items , which is not affordable for processing and had many redundant samples. Therefore in this step we seek to apply clustering technologies to reduce the size of these sets towards compactness. Compact sets in this case are the smallest sets that represent the wide variations of shapes for writing a word-part. Three techniques were applied to achieve these sets

- Hierarchical clustering. This step had been applied in the previous step to avoid shape redundancy when writing the same letter by all the different writers.
- Principal Component Analysis (PCA)
- K-Means Clustering.

Following are some details about the way these were used.

Let $S(\omega) = V_i(\omega)$ be a set of n vectors $V_i = (v_1, v_2, \cdots, v_{ni})$, representing the generated shapes for the word-part ω. To enable efficient and accurate processing, a preprocessing step is applies to uniformly simplify the stroke represented by V_i. Let us denote $V_i(\delta)$ the simplified vector, where δ is the error tolerance used to control the simplification. We define the feature α_i at the point p_i, on a given vector(point sequence), as the angle between the segment $\overline{p_i, p_{i+1}}$ and the x-axis. For each point vector $V_i(\delta)$ we generate a feature vector $F_i(\delta)$ using the feature α_i. In addition, to the vector $F_i(\delta)$ we use a parameter k that indicates the desired cardinality.

We first apply PCA on the covariance matrix of the n vectors F_i and use the k' eigenvectors derived from the largest k eigenvalue for dimensionality reduction of the original data. The original samples transformed by the k eigenvectors are clustered using the K-Means technique. Results transformed back to the original vectors are used as the k centroids to extract The representative vectors within each cluster. The result of the third step is a set of k vectors representing the k shapes in our desired compact set. The constants k an k' are fixed for each word-part as a percentage of the different shapes for each letter and the length of the word-part. In these clustering methods, we used the Euclidean distance to measure differences between shapes, which requires applying length normalization on the feature vectors F_i.

5. Experimental results

We have developed a system for Arabic online handwriting recognition to test the proposed synthetic database. This system was developed to enable online Arabic word-parts recognition based on elastic matching technique and geometric features. For comparison, we have used a dataset of five hundreds word-parts written by six different writers. Two hundred word-parts were selected carefully to represent many variations of writing styles, such as vertical and diagonal tiling (see Figure 1). The other 300, were selected randomly from the word-parts in the lexicon. The writers were advised to write each word-part several times to enable capturing their handwriting styles. The same set of five hundred word-parts was generated synthetically by our system.

The recognition rates when using our synthetic database as seen Table 1, are close to that obtained when using the manually generated database. To test the credibility of the concatenation process, two professional writers were asked to write the five hundred words and to generate one predefined font each. The same 500 word-parts were synthetically generated using the the generated fonts of each writer. In Table 2, we can see that the manually and the synthetic databases gives very close results. This proves that the concatenation process dose not negatively affect the ability of the synthetic database to cover the variety of the handwriting styles compared to manually generated one.

	Special Case		General Case	
	1 Of 1	1 Of 5	1 Of 1	1 Of 5
Synthetic database	82%	88%	82%	90%
Manual database	79%	89%	81%	91%

Table 1. Results of using the synthetic database compared to manually generated, first and five top ranked results. The special case columns stand for word-parts where letters does not tile horizontally.

	Special Case	General Case
Synthetic database	82%	80%
Manual database	85%	90%

Table 2. The recognition accuracy rates for synthetically and manually generated word-parts.

6. Conclusion and future work

We have presented an efficient novel approach for generating large datasets of word-parts for training and testing Arabic online handwriting recognition systems. Even though the results were very encouraging, we still seek to improve the clustering technique by using the Kohonen Self-Organizing Maps (SOM). We are considering using other metrics instead of the Euclidean such as the edit distance to measure differences between word-parts in the clustering stage. A major extension of this work is planned to enable generating databases for off-line and online Arabic script recognition using words and word-parts.

References

[1] Y. Al Ohali, M. Cheriet, and C. Suen. Databases for recognition of handwritten arabic cheques. 36(1):111–121, January 2003.

[2] H. Al-Yousefi and S. Udpa. Recognition of arabic characters. *IEEE Trans. Pattern Analysis Machine Intell*, 14(8):853–857., 1992.

[3] S. Alma'adeed. Recognition of off-line handwritten arabic words using neural network. In *GMAI '06: Proceedings of the conference on Geometric Modeling and Imaging*, pages 141–144, Washington, DC, USA, 2006. IEEE Computer Society.

[4] S. Alshebeili, A. Nabawi, and S. Mahmoud. Arabic character-recognition using 1-d slices of the character spectrum. *SP*, 56(1):59–75, January 1997.

[5] A. Amin. Off-line arabic character recognition : The state of the art. *Pattern Recognition*, 31(5):517–530, 1998.

[6] A. Amin and J. Mari. Machine recognition and correction of printed arabic text. *IEEE Trans. Syst. Man Cybern*, 19(5):1300–1306., 1989.

[7] S. El-Emami and M. Usher. On-line recognition of handwritten arabic characters. *IEEE Trans. Pattern Analysis Machine Intell*, 12(7):704–710., 1990.

[8] T. El-sheikh and R. Guindi. Automatic recognition of isolated arabic characters. *Signal Processing*, 14(2):177– 184., 1988.

[9] A. Gillies, E. Erl, J. Trenkle, and S. Schlosser. Arabic text recognition system. In *Proceedings of the Symposium on Document Image Understanding Technology*, 1999.

[10] S. Maddouri and H. Amiri. Combination of local and global vision modelling for arabic handwritten words recognition. In *Frontiers in Handwriting Recognition, 2002. Proceedings. Eighth International Workshop on*, pages 128–135, 2002.

[11] S. Mahmoud. Arabic character recognition using fourier descriptors and character contour encoding. *Pattern Recognition*, 27(6):815–824., 1994.

[12] N. Mezghani, A. Mitiche, and M. Cheriet. Bayes classification of online arabic characters by gibbs modeling of class conditional densities. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(7):1121–1131, 2008.

[13] R. W. N. Kharma, M. Ahmed. A new comprehensive database of hand-written arabic words, numbers and signatures used for ocr testing. In *IEEE Canadian Conference on Electrical and Computer Engineering*, pages 766–768, 1999.

[14] R. Saabni, F. Biadsy, J. El-Sana, and N. Habash. Segmentation-free online arabic handwriting recognition. *International Journal of Pattern Recognition*, page to appear, 2009.

[15] R. Saabni and J. El-Sana. Justifying holistic approach for arabic script recognition. Technical report, Ben Gurion University of the negev, Israel, 2008.

[16] S. T. Souici and L. M. Sellami. Off-line handwritten arabic character segmentation algorithm: Acsa. In *Eighth International Workshop on Frontiers in Handwriting Recognition*, pages 452–457, 2002.