Abstract

Lists of Figures

Abbreviation

Acknowledgements

1. Introduction
2. Background
   1. Characteristics of Arabic script
   2. Hand Writing Recognition
      1. Online vs. offline
      2. Writer dependent vs. Writer Independent
      3. Closed dictionary vs. open Dictionary
   3. Problems in the Arabic script
   4. Databases
3. Research Goals and Significance
4. Literature review
5. Our Approach
   1. The General Flow
   2. System Limitation
   3. Data Collection
   4. Data Preprocessing
   5. Learning Process
   6. Online Segmentation
   7. Letters Classification
   8. Word Part Classification
6. Results
7. Conclusions
   1. Contribution to the field
   2. Future Directions
8. Bibliography

# Abstract

Automated recognition of text has been an actively researched since the early days of computers. A 1972 survey cites nearly 130 works on the subject. [1]

Handwriting recognition is a task of transforming a language represented in its spatial form of graphical marks into its symbolic representation. Online Hand Writing Recognition refers to the situation where the recognition is performed concurrently to the writing process.

After a long period of focus on western and East Asian scripts there is now a general trend in the on-line handwriting recognition community to explore recognition of other scripts such as Arabic and various Indic scripts. One difficulty with the Arabic script is the number and position of diacritic marks associated to Arabic characters.

[Our Approach]

# Abbreviation

WP – Word Part - a single stroke connected component

DTW – Data Time Warping

EMD – Earth Movers Distance

PCA – Principle Component Analysis

LDA – Linear Discrimination Analysis

SVM – Support Vector Machine

# Introduction

Arabic handwritten script Recognition is drawing an increasing attention in recent years. The growing use of keyboard-less handheld devices provokes a transition in the Human-Computer data input interface. While in the past, keyboard and mouse devices were the primary mode of data entry, nowadays, hand and finger gestures are increasingly used for this task. Thus, a growing interest in online character recognition field has taken place.

Text recognition can be classified into two main fields: online and offline recognition. Online recognition peruses recognizing the text whilst being written. However, in the offline script recognition field, digital image containing text is fed to the computers and the system attempt to recognise the written text. [1]

The main existing approaches for script recognition are the holistic approach and the analytic approach. The holistic approach considers the global properties of the written text while the analytic approach involves segmentation and classification of each part of the text.

Automated script recognition of Latin, Chinese and Kanji has been a focus of study in the last decade and high recognition rates are reported. However, Arabic text recognition is at an early stage, a fortiori, when considering online handwriting recognition.

Arabic text, both handwritten and printed is cursive. Arabic word consists of one or more sub-word (which we will refer as word parts). Each sub word consists of one character or more. Thus most of the online Arabic recognition systems perform the recognition soon after the word part is scribing is completed using holistic or analytic approach.

In the present work we propose a novel approach which combines both holistic and analytic approaches for recognizing Arabic online script. In our approach, the recognition process is performed whilst the word part is being written. Using flexible letters segmentation algorithm in the first stage; letters candidates’ selection by using agile but loose classifier in the next stage; and then a holistic word part recognition using costly yet accurate classifier.

# Background

## Online Hand Writing Recognition

There is a vast transition in the field of personal computers from the desktop to handheld devices; thereby single hand entry methods are more suited to this shift than a keyboard. Pen or even figures or hand gestures are more natural and convenient interface.

The large number of writing styles and the variability between them makes the problem of writer-independent unconstrained handwriting recognition a very challenging pattern recognition problem.

Nowadays, the primary mode of data input from a human to a computer is still the keyboard. However, the amount of information processed by computers is rapidly increasing. Given this, the time consumption of information exchange between human and computers is becoming a serious bottleneck. In order to be effective, the user interface has to be both effective and natural. Thereby, requiring no learning curve to the user. [2]

[Talk about smartphones/Tablets/etc.]

While much of the today’s data is directly entered into computers using the keyboard, many tasks still exist in which people tend to prefer handwriting over keyboard entry. Note taking (e.g. in classrooms) is a task that can still be done more efficiently by hand for most users. In addition, while people can produce annotated hand sketches very quickly, data entry into a computer using a combination of the mouse and keyboard is relatively time consuming. [2]

Smartphones and tablets are pocket sized consumer devices that can store calendars and address books, provide access to emails, the web, and contain other productivity tools. These devices are too small to have full sized keyboards, or sometimes may be too small for any keyboard at all, requiring pen, hand gestures, figure gestures or voice interface to enter data. [2]

The problem of handwriting recognition has now been a topic of research for over four decades. There are many types of problems (with varying complexity) within handwriting recognition, based on how the data is presented to the recognition system, at what level the data can be unambiguously broke n into pieces (e.g. individual characters or words), and the transcription complexity of the language used. [2]

At the highest level. Handwriting recognition can be broken into two categories: offline and online. Offline handwriting recognition focuses on documents that have been written on paper at some previous point of time. Information is presented to the system in the form of scanned image of the paper document. In contrast, online handwriting recognition focuses on tasks where recognition needs to be performed at the time of writing. This requires the use of special equipment, such touch screen or digitizing tablet, to capture the strokes of the pen as that are being written. The trace of a writer’s pen is stored as a sequence of points sampled at equally spaced time intervals. The information captured for each sample is the  coordinates. While this sequence can be used to construct a static image of the writing, thus allowing offline character recognition techniques to be applied, it has been shown [63] that the information about the pen dynamics can be used to obtain a better recognition accuracies than the static data alone. Therefore, it is beneficial to capture the data in an online form, even if the real-time processing requirements can be relaxed. [2]

Another advantage of online handwritten data over offline data is the availability of the stroke segmentation and the order of writing. Ink in static images must first be separated from the image background, creating a potential source of error. The ability to detect the states of “pen-down” (when the pen touches the tablet or the finger touches the touch screen) and “pen-up” can also be used. A single stroke is defined as the sequence of sample points occurring between consecutive pen-down and pen-up transitions. However, a complication occurs when a stroke is added to a character in a word after the rest of the word has already been written, such as the cross of a ‘t’ or an ‘x’, or the dot of an ‘i’ or a ‘j’. These types are called delayed strokes. [2]

## Characteristics of Arabic Script

[See Online handwriting recognition for the Arabic Letter Set]

The Arabic Aleph bet is widely used for more than twenty different languages such as Farsi, Urdu, Malay, Housa and Ottoman Turkish. Arabic is used in over 20 different countries, written by more than 100 million people and spoken by 234 million people. Although the spoken Arabic is slightly different from country to country, the written Arabic is standard system used all over the Arab world. [3] [4]

Arabic Scripts consists of 28 basic letters, 12 additional special letters, and 8 diacritics. Arabic script is written from right to left in a semi-cursive manner in both printed and handwritten. Most letters are written in four different letter shapes depending on their position in a word, e.g., the letter ع (Ain) appears as ع (isolated), عـ(initial), ـعـ (medial) and ـع (final). Among the basic letters, six are Disconnective – ا (Alef), د (Dal), ذ (Thal), ر (Reh), ز (Zain) and و (Waw). Disconnective letters do not connect to the following letter and have only two shapes each. The presence of this letters interrupts the continuity of the graphic form of a word. We denote connected letters in a word, as word-part. If the word-part is composed of only one letter, this letter will be in its isolated shape. [5]

Certain characteristics relating to the obligatory dots and strokes of the Arabic script distinguish it from Roman script, making the recognition of words in Arabic more difficult than in Roman script. First, Most Arabic letters contain dots in addition to the letter body, such as ش (Sheen) which consists of س (Seen) body and three dots above it. In addition to dots, there are stroke that can attach to a letter body creating new letter such as ك, ط and لا. These dots and strokes are called *delayed strokes* since they are usually drawn last in the in handwritten word-part/word. Second, eliminating, adding or moving a dot or stroke could produce a completely different letter and, as a result, produce a word other than the one that was intended (see Table 1). Third, the number of possible variations of delayed strokes is greater than those in Roman scripts, as shown in Figure 2. There are only three such strokes used for English: the cross in the letter *t*, the slash in *x*, and the dots in *i* and *j*. Finally, in Arabic script a top-down writing style called *vertical ligatures* is very common – letters in a word may be written above their consequent letters. In this style, the position of letters cannot be predefined relative to the baseline of the word. [5]

Saabni and El-sana have explored a large collection of Arabic texts and extracted 300,000 different word combinations of 82,000 different word-parts. Ignoring the additional strokes reduced the number of different word-parts to 40,000. [3]

### System Limitation

As mentioned before, Arabic is a cursive written language and it contains about 40k possible word parts. By “*possible”*, we mean that there is an Arabic word which contains the word part. Arabic letters may differ by additional stroke above or beneath the main stroke. For example, the Arabic letter ف (Fa) contains a single dot above the main stroke, however the letter ق (Qa), contains double dots, both having identical main body. In our work we recognize and classify the main body of the letter and ignore the additional stroke entirely. As a result, the number of different letters drops from 29 to18 and the number of different possible word parts decreases to []. It is important to comment that taking the additional strokes into consideration may be exploited to boost the classification rate.

The main body of most Arabic letters is written by a single stroke. However, there are some letters that usually written using 2 strokes, such as the letter ـكـ which is the middle \_\_ of the letter ك (Ka). The writer usually writes ـلـ and adds the final upper slanted line when the main body is completed, as if he writes an additional stroke.

Another problem arises when trying to recognize Arabic transcript, is that, different writers may write the main body of the same word part in a different number of strokes. For instance, the main body of the word part بىت (Bayt - Home), is usually written in a single stroke however, sometimes it may be written by some writers using 3 strokes.

We have also considered the common combination of the letter ل followed by the vowel ا as a single letter which is commonly drawn as لا or لا.

For these mentioned complexities, when recognizing Arabic scripts, most researches have preferred the holistic approach.

## The ADAB Database

The database ADAB was developed to advance the research and development of Arabic on-line handwritten text recognition systems. This database is developed in cooperation between the Institutfuer Nachrichtentechnik (IfN) and the Research group on intelligent Machines (REGIM). The database consists of 20575 Arabic words handwritten by more than 170 different writers, most of them selected from the narrower range of the National school of Engineering of Sfax (ENIS). The ADAB database is freely available for non-commercial research (www.regim.org). Our aim was to collect a database of handwritten town names written in a similar quality as on a Mobile Phone with a digital input device. The collection process starts when the writer clicks on start bottom. The collection tool generates a town name randomly from 937 Tunisian town/village names, the writer must write the displayed word as it is shown in Fig.3. A pre-label will be automatically assigned to each file. It consists of the postcode in a sequence of Numeric Character References which will be stored in the UPX file format. An InkML file including trajectory information and a plot image of the word trajectory are also generated. Additional information about the writer can also be provided. The ADAB-database is divided to 6 sets. Details about the number of files, words, characters, and writers for each set 1 to 6 are shown in the table below.(Kherallah, Tagougui, Alimi, Abed, & Margner, 2011)

# Literature Review

Automated recognition of text has been an active subject of research since the early days of computers. A 1972 survey cites nearly 130 works on the subject [6]. Despite the age of the subject, it remains one of the most challenging and exciting areas of research in computer science. In recent years it has grown into a mature discipline, producing a huge body of work.

Despite long standing predictions that handwriting, and even paper itself, would become obsolete in the age of the digital computer, both persist. Whilst the computer has hugely simplified the process of producing printed documents, the convenience of a pen and paper still makes it the natural medium for many important tasks.

A brief survey of students in any lecture theatre will confirm the dominance of handwritten notes over those typing on laptops. However, the ease and convenience of having information in digital form provides a powerful incentive to find a way of quickly converting handwritten text into its digital equivalent.

The field of handwriting recognition can be split into two different approaches. The first of these, on-line, deals with the recognition of handwriting captured by a tablet or similar touch-sensitive device, and uses the digitised trace of the pen to recognise the symbol. In this instance the recogniser will have access to the x and y coordinates as a function of time, and thus has temporal information about how the symbol was formed. The second approach concentrates on the recognition of handwriting in the form of an image, and is termed off-line. In this instance only the completed character or word is available. It is this off-line approach that will be taken in this report.

# Our Approach

While the holistic approach is a common approach to recognize handwriting both in online and offline domains, this approach is not practical in Arabic script which has a large set of vocabulary, since it needs to train the system for each word in the dictionary. [1] We suggest recognizing the Arabic scribing in a progressive manner in which we try to segment the handwriting and extract the letters from the being written sequence data and recognize the word part being written in an analytic approach. One of the significant benefits of this approach is that, the magnitude of the classes the recognition algorithm is trying to match is much smaller. As was mentioned before, there are about 40,000 valid word parts, with the different main strokes, there are much more if we take into consideration the additional strokes, therefore, the samples space is very large. However, when trying to identify letters, the learning domain is much smaller as there are a small set of letters, thus a better recognition can be performed.

## Data Collection and Processing

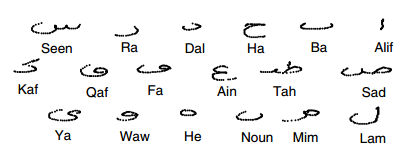
The following diagram describes the letters learning process of our approach.



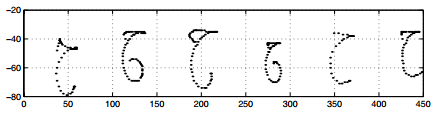
### Letters samples collection

A most important element in developing a learning system is the data, which is usually insufficient. Thus, the data collection stage is an important stage in the development of our system.

The Arabic alphabet consists of 29 isolated characters. If we ignore all diacritical marks, which do not carry important information about shape, we obtain 18 shapes (Fig.4)



The writing of these characters was not constrained, leading to a wide variety of size, and orientation. Figure (5) shows some samples of the “Ha” letter written by several writers.



We utilize two sources of letters samples. The first is a self-made database of letters that were collected from different age Arabic writers using digitizer tablet. We have developed an application, using Matlab, which asks the user to draw a letter on an electronic pad, the sequence data of the letter shape was saved as a file on the file system. The rectangle in which the user was asked to draw the letter is 1X1. See image of the GUI below. [Add an Image of the Matlab data collection UI]. The other source was the ADAB database

## Recognition Methodology Overview

Our approach combines both Analytic and Holistic approaches. A main stages of the whole word part recognition process is seen in chart below.



Figure 1 – The main stages of the system recognition.

At the first stage the system nominate demarcation points, in the next stage, in an offline process, the system determine the best Segmentation of the word-part. At the third stage for each sub-sequence, recognition candidates are selected for each subsequence, then the system generate all possible word-parts based on the candidate letters, and embed it to some space. In the last stage, a final word-part is selected.

## Demarcation Points Nomination

Demarcation point (sometimes referred as *Segmentation point*) is a junction point in the written stroke that is a vertical handler between letters in a word-part. Several segmentation techniques have been proposed in the literature for Arabic OCR. However correct and efficient segmentation of Arabic text is still considered a challenging and a fundamental problem even for offline printed text. Performing such task in an online manner for handwritten texts when the segmentation is being done while the word is being written is even a more challenging. Over segmentation and under-segmentation are the main problems such algorithms are facing. Later we will show how this problem is handled by our method.

This stage is performed while the word part is being written. The goal is to be able to decide effectively if a certain point is a demarcation point with high probability. While the word part is being scribed, our system tries to segment the sequence by identifying candidate demarcation points. To identify segmentation points, attributes of Arabic segmentation points need to be learned. To identify demarcation point we have used the following 2 characteristic: 1. Demarcation point lives in a horizontal segment. 2. Demarcation point lives in a forward Segment. We use SVM identify demarcation point using these two features.

The under-segmentation problem is usually as a result of two letters combinations that doesn’t contain a horizontal handler between them. For example, the following word-part لم (Lam - Not) which is a combination of the letters ل (L) and م (M). Our approach overcomes this problem by broaden the set of letters classes to include such combinations. We will refer to such combinations as hyper-letters. By this, even if the segmentation process did not identify a demarcation point between this two letter, it will be recognized in the later process as a single letter.

[Read the paper about segmentation]

## Segmentation selection and Letter classification

Good segmentation will outcome a better letter recognition. The goal of this stage is to overcome the over-segmentation problem. By selecting the set of demarcation points which gives the best letter recognition results. This is done after the writer has completed scribing the main body of the word-part. In this stage the system selects the best segmentation of the word-part sequence. Clearly, Segmentation induces partition to subsequences. Where each subsequent is geometric of a letter or a hyper-letter thus is classified to a set of 3 candidate hyper-letters, the evaluation score of the best classification is taken to represent the quality of the subsequent segmentation. This scoring is then given as an input to a dynamic programming algorithm to select the best segmentation. The classification algorithm we use in this stage to classify the subsequences to letters is Multiclass SVM, Taking into consideration the position of the letter.

## Candidate Word-Part generation and embedding

After we have determined the segmentation, and have classified each sub-sequent to a small set of letters, the system generate all possible word-parts sequences using the samples of letters in our letters database using the most common ways to write the letter. Here we use the Holistic approach to classify the word-part. However, instead of taking the whole dictionary of possible word-parts, our space of candidate word part will be the set of word parts taken from the previous stage. Doing that, we extract the features of the all possible word-parts and embed them using approximate EMD approach. EMD is Euclidean metric, thus we are able to use LSH to build a fast retrieval data structure.

## Word-Part classification

At this stage the features of the written sequence are extracted. Embedded to the Candidate word-part space. The classification is the nearest neighbor in the LSH.

### Letters Pre-Processing

In this stage we reduce the data variation of the letters data as much as possible, which is caused by the uniform temporal sampling of the digitizer.

Digitizers often oversample slow pen motion regions and under-sample fast motion regions. [1]

This phase contains 2 main parts: *Simplification* and *Resampling*.

#### Simplification

In order to eliminate redundant points irrelevant for pattern classification and screening out unwanted noise and vibrations in the letter inscription we have used the Douglas-Peucker Polyline Simplification algorithm described in [18].

Briefly, it is a line simplification algorithm to reduce the number of vertices in a piecewise linear curve according to a specified tolerance. The algorithm is also known as Iterative Endpoint Fit.

Assuming the stroke presentation is a sequence of points, the sensitivity parameter that was used in our work was, in most cases,, when is defined and .

#### Resampling

The resampling is needed to prevent the unbalanced sampling density, which may be influenced by the sampling rate and the user non-uniform letter drawing.

## Arabic Progressive Recognition

The process of learning letters in different positions is described below.

1. Segmentation – holistically, segment the given data into sub-sequences that represent letters. This is done by spotting out potential segmentation points.
2. Recognition – Recognizing the given sub-sequences of the Segmentation phase and classify what letter was just completed.

## Letters Learning

The general process of the letter learning is described in Figure [Fig no.]

### Demarcation Points

## Feature Extraction

## Features Transformation and Dimensionality Reductions methods

Feature transformation is a group of methods that create new features (predictor variables). The methods are useful for dimension reduction when the transformed features have a descriptive power that is more easily ordered than the original features. There are 2 main approaches for this task. One is *feature selection* and the other *is feature transformation*.

Dimensionality Reduction is a process of reducing the number of random variables taken into consideration in the learning and classification of Data. Ideally, the reduced representation should have a dimensionality that corresponds to the intrinsic dimensionality of the data. Reducing the dimensionality of the features vectors would not only simplify and rapid the learning and classification task but rather boosts the classification accuracy.

Feature transformation technique is much more suitable to be implemented in our approach.

In this work we have chosen to use two techniques applied in sequential manner in order to obtain the most efficient and linearly discriminative components, *Principle Component Analysis* (PCA) and *Linear Discrimination Analysis* (LDA) Technique.

Why we have used both and what give us every method and how we did join them together to get the most out of both.

### Combining PCA and LDA

Even though LDA is preferred in many application of dimension reduction, it does not always outperform PCA. In order to optimize discrimination performance in a more generative way, a hybrid dimension reduction model combining PCA and LDA is used in this work.

In our approach, the dimensionality reduction process can be outlines as follows: the pre-processed feature matrix M projected into subspace  using PCA and then into the subspace  using LDA. In the PCA stage, the largest t eigenvalues are selected to create the PCA projection matrix. t is the number of eigenvalues which guarantee energy E is greater than 0.98. The data preservation value is calculates as:

|  |  |
| --- | --- |
|  | Equation 9 |

Where is the i-th eigenvalue.

The dimensionality of is much smaller that the dimensionality of M. At the second phase LDA is used to project to . The dimension of subspace is smaller than the sunspace by 1.

Recognition

Feature Work

1. Incremental progressive handwriting recognition.

# Bibliography

|  |  |
| --- | --- |
| [1] | M. Al-Ammar, R. Al-Majed and H. Aboalsamh, "Online Handwriting Recognition for the Arabic Letter Set," *Recent Researches in Communications and IT.* |
| [2] | S. D. Connell, "Online Handwriting Recognition Using Multiple Pattern Class Models," Michigan, 2000. |
| [3] | R. Saabni and J. El-sana, "Efficient Generation of Comprehensive Database from Online Arabic Script Recognition". |
| [4] | I. A. Jannoud, "Automatic Arabic Hand Written Text Recognition System," *American Journal of Applied Sciences,* pp. 857-864, 2007. |
| [5] | F. Biadsy, R. Saabne and J. El-Sana, "Segmentation-Free Online Arabic Handwriting Recognition," 2010. |
| [6] | L. Harmon, "Automatic recognition of print and script," *Proceedings of the IEEE,* vol. 60, no. 10, pp. 1165 - 1176 , 1972. |
| [7] | P. Senin, "Dynamic Time Warping Algorithm Review," Honolulu, USA, 2008. |
| [8] | T. M. Rath and M. Manmatha, "Word Image Matching Using Dynamic Time Warping". |
| [9] | S. Salvador and P. Chan, "FastDTW: Toward Accurate Dynamic Time Warping in Linear Time and Space," Melbourne. |
| [10] | C. A. Ratanamahatana and E. Keogh, "Three Myths about Dynamic Time Warping Data Mining". |
| [11] | I. Poitr and N. Thamper, "Fast Image Retrieval via Embeddings," Third International workshop on Statistical and Computional Theories of Vision, Nice, France, 2003. |
| [12] | "The Earth Mover's Distance," The University of EDINBURGH, [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL\_COPIES/RUBNER/emd.htm. [Accessed 12 April 2012]. |
| [13] | S. Shirdhonkar and D. W. Jacobs, "Approximate earth mover's distance in linear time," University of Maryland, Maryland, 2008. |
| [14] | S. Belongie, J. Malik and J. Puzicha, "Shape Matching and Object Recognition Using Shape Contexts," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 24, p. 509–521, 2002. |
| [15] | R. Saabni and A. Bronstein, "The Multi Angular Descriptor: new binary and gray images descriptor for shape recognition". |
| [16] | J. Sadri, C. Y.Suem and T. D. Bui, "Application of Support Vector Machines for Recognition of Handwritten Arabic/Parsian Digits". |
| [17] | C. Bahlmann, B. Haasdonk and H. Burkhardt, "Online Handwriting Recognition with Support Vector Machines - A Kernel Approach," *Workshop on Frontiers in Handwriting Recognition (IWFHR),* pp. 49-54, 2002. |
| [18] | D. Douglas and T. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *The Canadian Cartographer,* 1973. |