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. Online Hand Writing Recognition
   4. Databases
3. Research Goals and Significance
4. Literature review
5. Our Approach
   1. General Description of the system (Flow)
   2. Data Collection
   3. Data Preprocessing
   4. Feature Extraction
   5. Metrics Distances
      1. DTW
      2. ERP
      3. EMD
   6. Features
      1. Shape Context
      2. Multi Angular Descriptor
   7. Fast Retrieval data structure
      1. Kdtree
      2. LSH
   8. Dimensionality Reduction
      1. PCA
      2. LDA
   9. Learning Models
      1. SVM
      2. WaldBoosting
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]

## 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.

## Dynamic Time Warping

***Dynamic time warping*** (DTW) is an algorithm for measuring similarity between two time serieses. The method is to find an optimal alignment between two given sequences. Intuitively, the sequences are warped in a non-linear fashion to match each other. DTW solves the discrepancy between intuition and calculated distance by achieving the optimal alignment between sample points of two times sequences. Restrictions can be imposed on the matching technique to improve the matching results and the algorithm’s time complexity. The only restriction placed on the data sequences is that they should be sampled at equidistant points in time which can be easily achieved by re-sampling. A naïve approach to calculate the distance between two time serieses can be to resample one of the serieses and to compute the distance sample by sample, the problem with this technique is that it does not yield intuitive results as it might match samples that do not correspond well. See figure 1.

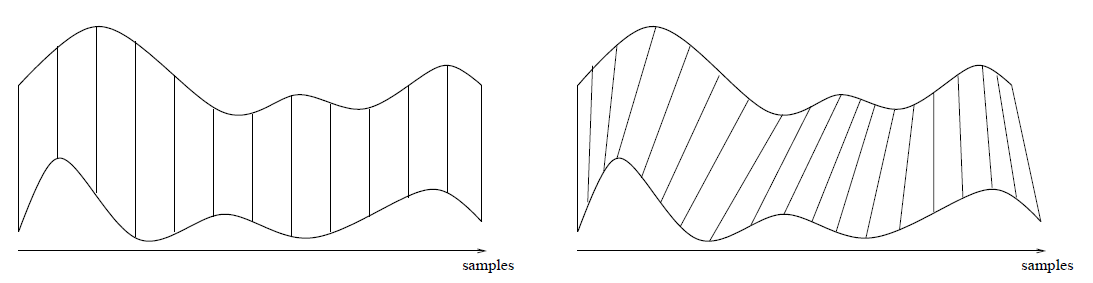


Figure 1 – The right scheme shows sample-by-sample naïve alignment after resampling and in the left scheme the alignment was performed using DTW

Formally, given two time serieses, and , DTW yields an optimal warping path, where  for which satisfies the two following:

1. **Start and End point constrain:** and.
2. **Time preservation of the points:** for each two points in W,,  and the following holds: and .
3. **Local continuity constrain:**

The Distance of the warping path is defined as follows:

|  |  |  |
| --- | --- | --- |
|  |  | Equation 1 |

 is the distance of the two data point indexes (one from X and the other from Y) in the kth element of the warping path.

The warping path which has the minimal  associated with alignment is the optimal warping path which we have previously denoted by.

Using dynamic programing approach, DTW yields the optimal warping path by constructing the **accumulated distance matrix.** We will denote this matrix by the letter, where is the minimum distance warping path that can be constructed from the two time serieses and. Wherefore the value in  will contain the minimum-distance warping path between  and .

The accumulated distance matrix is calculated as follows:

|  |  |  |
| --- | --- | --- |
|  |  | Equation 2 |
|  |  |  |
|  |  | Equation 3 |

Once matrix D is calculated the optimal warping path  is retrieved by backtracking the matrix  from the point  to the point following the greedy strategy of looking for the direction from which the current distance is taken.

It is easy to note that the time and space complexity of DTW is. [7] [8] [9]

### DTW Speedup

One drawback of DTW is its quadratic time and space complexity, thus, many speedup methods have evolved. These methods can be fall in three categories:

1. Constrains: Limiting the amount of calculated cells in the cost matrix.
2. Data Abstraction: Running the DTW on an abstract representation of the data.
3. Lower Bounding: Reducing the times DTW need to run when clustering and classifying time serieses.

Constrains: The most popular two constrains on DTW are the Sakoe-Chuba Band and he Itakura parallelogram, which are shown in figure 2. The grayed out area is the cells of the cost matrix that are filled by the DTW algorithm for each constrain. The warping path is looked for in the constrain window. The width of the window is specified by a parameter. By using such constrains the speedup factor is a constant and the DTW complexity is still (where M and N are the time serieses lengths). Note that if the warping path is does not reside in the constrain window, it will not be found by DTW, thus, such method is usually used when the warping path is expected to be in the constrain window.

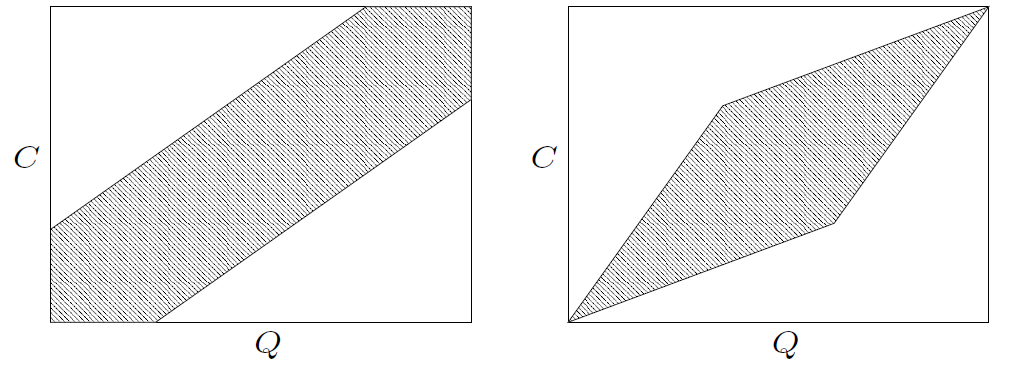


Figure 2 – Cost matrix constrains: Sukoe-Chuba Band (left) and the Itakura Parallelogram (right).

Data Abstraction: Speedup using data abstraction is performed by running DTW on a reduced presentation of the data thus reducing the cell numbers that need to be computed. The warp path is calculated on the reduced resolution matrix and mapped back to the original (full) cost matrix.

Lower bounding: Searching the most similar time series in the database given a template time series can be done more efficiently using lower bound functions than using DTW to compare the template to every series in the database. A lower-bounding is cheap and approximate. However, it underestimates the actual cost determined by DTW. It is used to avoid comparing serieses by DTW when the lower-bounding estimate indicates that the time series is worse match than the current best match.

FastDTW, proposed by Stan Salvador and Philip Chan, approximate DTW in a linear time using multilevel approach that recursively projects a warping path from the coarser resolution to the current resolution and refines it. This approach is an order of magnitude faster than DTW, and also compliments existing indexing methods that speedup time series similarity search and classification. [9]

Ratanamahatana and Kough have shown the following fact about DTW in their work in [10]: Comparing sequences of different lengths and reinterpolating them to equal length produce no statistically significant difference in accuracy or precision/recall.

Although in our work we do make reinterpolating of sequeces to compare equilenth sequences.

## Earth Movers Distance

Earth movers distance (EMD) is a measure of the dissimilarity between two histograms. Descriptively, if the histograms are interpreted as two different ways of piling up a certain amount of dirt, the EMD is the minimal cost of turning one pile to other, where the cost is assumed to be the amount of dirt moved times the distance by which it moved.

EMD has been experimentally verified to capture well the perceptual notion of a difference between images. [11]

[Add a picture]

Computing EMD is based on a solution to the well-known transportation problem. It can be computed as the minimal value of a linear program.

[EMD formal definition]

EMD naturally extends the notion of a distance between single elements to that of a distance between sets, or distributions, of elements. In addition, it’s a true metric if the signatures are equal. [12] A major hurdle to using EMD is its computational complexity (for an N-bin histogram).Various approximation algorithms have been proposed to speed up the computation of EMD.

A histogram can be presented by its signature. The signature of a histogram is a set of clusters where each cluster is represented by its mean (or mode), and by the fraction of the histogram that belongs to that cluster.

### EMD Embedding

The Idea of the Embedding is to compute and concatenate several weighted histograms of decreasing resolutions for a given point set. The embedding of EMD given in [11] provides a way to map weighted point sets A and B from the metric space into the normed space , such that the  distance between the resulting embedded vectors is comparable to the EMD distance between A and B. The motivation of doing so is that the working with normed space is desirable to enable fast approximate Nearest Neighbors (NN) search techniques such as LSH.

[Embedding formal definition]

### Wavelet EMD

A work conducted by Shirdhonkar and Jacobs in [13] has proposed a method for approximating the EMD between two histograms using a new on the weighted wavelet coefficients of the difference histogram. They have proven the ratio of EMD to wavelet EMD is bounded by constants. The wavelet EMD metric can be computed in time complexity.

[Embedding formal definition]

## Features

The selection of valuable features is crucial in pattern recognition.

## Shape context

S. Belongie et al. have defined in [14], a point matching approach named Shape Context. The Shape context is a shape matching approach that intended to be a way of describing shapes that allows for measuring shape similarity and the recovering of point correspondences. This approach is based on the following descriptor: Pick n points on the shape’s contour, for each point  on the shape, consider the  other points and calculate the coarse histogram of the relative coordinates such that is defined to be the shape context of. The bins are normally taken to be uniform log-polar space. This distribution over relative positions is robust and compact, yet highly discriminative descriptor.

To match two points, from the first shape and from the other shape, define  denote the cost of matching, these two points, use the  test statistic:

|  |  |
| --- | --- |
|  | Equation 4 |

.

To perform a one-to-one matching that matches each point  on shape 1 and  on shape 2 that minimizes the total cost of matching: is needed.

This can be calculated in  time using the Hungarian method.

[Add a picture]

## Multi Angular Descriptor

The Multi Angular Descriptor (MAD) is a shape recognition method described in [15], which captures the angular view to multi resolution rings in different heights. Given an Image I of a connected component, calculate the centroid C and the diameter D of the image I. draw a set of rings centered by C with different radius values which are derived from the diameter D. draw k points on each ring taken with uniform distance from each other. Each point in each ring serves as an upper view point watching each contour point in the shape.

## Fast retrieval data structures

Now we will give an overview on 2 methods that tries to answer the second question.

### K dimensional tree

## Support Vector machine

SVM (Support Vector Machine) is a powerful tool for classification developed by Vapnik in 1992. It has originally been proposed for two-class classiﬁcation; however, it can be easily extended to solve the problem of multiclass classification. In this work we use one-to-one/ one-to-many method to do so. SVM is widely used in object detection & recognition, content-based image retrieval, text recognition, biometrics, speech recognition, etc. Without Losing generality, we consider the 2 class classification problem. We give a brief introduction for the SVMs basics. SVM paradigm has a nice geometrical interpretation of discriminating one class from the other by a separating hyperplane with maximum margin. SVM can be altered to perform nonlinear classification using what is called the kernel trick by implicitly mapping the inputs into high-dimensional feature spaces. Given a training sample set  where a d-dimensional sample,  is the class label and N is the size of the training set.

Support vector machine first map the data from the sample from the input space to a very high dimensional Hilbet Space.. The mapping  is implemented implicitly by a kernel function  that satisfies Mercer’s conditions. Kernels are functions that give the inner product of each pair of vectors in the feature space, such that. Then in the high dimensional feature space H, we try to find the hyperplane that maximizes the margin between the two classes, and minimizing the number of training error.

The decision function is as follows:

|  |  |
| --- | --- |
|  | Equation 4 |

Where

|  |  |
| --- | --- |
| And | Equation 5 |

Training as SVM is to find  that minimizes the following quadratic cost:

|  |  |
| --- | --- |
|  | Equation 6 |

Where  is a parameter chosen by the user, a large  corresponds to a higher penalty allocated to the training errors. This optimization problem can be solved using quadratic programming. [16]

Many implementation of kernels have been proposed, one popular example is the Gaussian Kernel:

|  |  |
| --- | --- |
|  | Equation 7 |
|  |  |

[Add a picture]

## Gausian dynamic time warping using DTW as a metric

Claus Bahlmann et al. in [17] have proposed combining DTW and SVMs by establishing a new kernel which they have named Gaussian DTW (GDTW) kernel. They have achieved, by incorporating DTW in the Gaussian kernel function, a method to handle online handwriting input which is a variable-sized sequential data. Clearly, the assimilation of the DTW in the Gaussian kernel will result the following kernel function

|  |  |
| --- | --- |
|  | Equation 8 |
|  |  |

We should note that the DTW is not a metric, as the triangle inequality do not hold in many cases, thus the GDTW may miss some important properties as satisfying the Mercer’s condition. This would imply that the optimization problem in 6 may not convergent to the global minimum.

# Our Approach

Recognition of Arabic transcription follows 2 approaches. An analytic approach, wherein the word or word parts are segmented into letters, Recognized and then reassembled into a whole word and a holistic approach where no segmentation is needed, and the words or word-parts are recognized as a whole.

## Our Approach for Arabic Word Parts recognition

As we have mentioned in the section about the Arabic script, 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. We use the holistic approach to perform the recognition. As have been mentioned earlier, 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 try to recognize and classify the main body of the letter thus have ignored the additional stroke entirely. As a result, the number of different letters drops to [] and the number of different possible word parts decreases to []. It is important to comment that taking the additional strokes into consideration may simplify the classification task.

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 ك (Ka) when written in the middle of a word part which look like that ـكـ. 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), may be written by some writers in a single stroke and by other writer using 3 strokes. As a result, we can’t count on the reasoning, that if the user rises his hand from the writing pad, it means that he has finished to write a word part.

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.

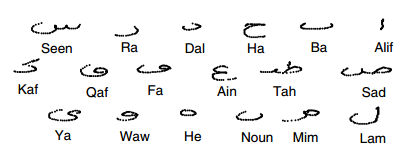
The following diagram describes the learning process of our approach.



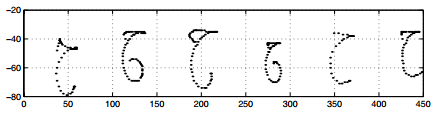
### Letter samples collection

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 utilized two sources of letters samples. The first was 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 [citation needed]. [Write about ADAB and our processing]

### 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.

* 1. A complete WordPart database
  2. Additional Processing on the wordparts
  3. Feature Extraction
     + Sequence
     + Shape Context
     + MAD
  4. EMD Embedding
  5. Dimensionality Reduction
     + PCA
     + LDA
  6. Fast retrieval database
     + Kdtree
     + LSH
* The recognition Process is similar to the learning Process however after we get the processed output, we query the database for the most similar object

## Arabic Progressive Recognition

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. We compare several method of learning using the Support Vector Machines (SVMs) with different kernel’s selection.

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.]



Figure 3 - Letters Learning Process

### Demarcation Points

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. [add citation] 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 text when the segmentation is being done while the word is being written, is even a more tantalizing.

Over segmentation and under-segmentation are the main problems such algorithms are facing.

While the word part is being scribed, our system tries to segment the points 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 these SVM for demarcation point identification.

[Read the paper about segmentation]

## 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.

### Principle Component Analysis

PCA was invented in 1901 by Karl Pearson. It is a linear technique for dimensionality reduction, which means that it performs dimensionality reduction by embedding the data into a linear subspace of lower dimensionality. Although there exist various techniques to do so, PCA is by far the most popular (unsupervised) linear technique. PCA is an orthogonal linear transformation that transforms the data to a new coordinate system such that the greatest variance by any projection of the data comes to lie on the first coordinate (names the first principal component), the second greatest variance on the second coordinate, and so on. Each principal component is a linear combination of the original variables. All the principal components are orthogonal to each other, so there is no redundant information. The principal components as a whole form an orthogonal basis for the space of the data.

The full set of principal components is as large as the original set of variables. But taking the first few principal components will preserve most of the information in the data, and reduces the data dimensions.

### Linear Discrimination Analysis

PCA is an unsupervised technique and as such does not include label information of the data. The following example demonstrates the problem drawback: Imagine 2 cigar like clusters in 2 dimensions, one cigar has y = 1 and the other y = -1. The cigars are positioned in parallel and very closely together, such that the variance in the total data-set, ignoring the labels, is in the direction of the cigars. For classiﬁcation, this would be a terrible projection, because all labels get evenly mixed and we destroy the useful information. A much more useful projection is orthogonal to the cigars, i.e. in the direction of least overall variance, which would perfectly separate the data-cases.

LDA is closely related to PCA in that they both look for linear combination of variables which best explain the data. LDA explicitly attempts to model the difference between the classes of data. In this method, variability among the feature vectors of the same class is minimised and the variability among the feature vectors of different classes is maximised.

The LDA performs dimensionality reduction while preserving as much of the class discriminatory information as possible. Without going into the math, in order to find a good projection vector, we need to define a measure of separation between the projections. The solution proposed by Fisher is to maximize a function that represents the difference between the means, normalized by a measure of the within-class scatter.

Although, LDA assumes that the distribution of samples in each class is Gaussian and that we cannot prove that the handwritten letters are distributes in a Gaussian manner, we selected LDA as

[put the picture that demonstrates the problem with PCA]

[Should I Add mathematical explanation?]

### 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. |