Abstract

Lists of Figures

Acknowledgements

1. Introduction
   1. Online Hand Writing Recognition
   2. Characteristics of Arabic script
   3. Previous work
      1. An overview of Online Handwriting recognition
   4. Database
   5. Outline of forthcoming chapters
2. Background / Preliminaries
   1. Metrics
      1. DTW
      2. EMD
   2. Features
      1. Shape Context
      2. Multi Resolution Shape Context
   3. Fast Retrieval data structure
      1. Kdtree
      2. LSH
   4. EMD embedding
   5. PCA
   6. LDA
   7. Nearest Neighbors
      1. Most Centrally object
3. Phase 1: Offline Handwriting recognition
   1. Implementation
   2. Results
   3. Discussion
4. Phase 2: Progressive/Just-In-Time (Dynamic) Arabic Online Handwriting Recognition
5. Discussion
   1. Introduction
   2. Summary of results
   3. Contribution to the field
   4. Conclusions
   5. Future Directions
6. Bibliography

# Introduction

## Online Hand Writing Recognition

The field of personal computing has begun to make a transition from the desktop to handheld devices, thereby requiring input paradigms that are more suited for single hand entry than a keyboard. Data entry using a pen or even by using the finger forms a natural, 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. [1]

[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. [1]

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. [1]

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. [1]

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. [1]

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. [1]

## Characteristics of Arabic Script

The Arabic Aleph bet is widely used for more than twenty different languages such as Farsi, Urdu, Malay, Housa and Ottoman Turkish. [2]

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. [3]

The Arabic script is different from the western scripts in that it combines letters into words. [2]

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. [3]

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. [2]

# Preliminaries

Sequence retrieval from large databases is a problem of interest in vision and database communities.

The central questions in this area are:

1. How to design a (dis)-similarity measure that quantifies the perceptual of two sequences being similar.
2. How to build a data structure that quickly identifies the closest sequence from the databases.

In the following will survey two methods that attack the first question and 2 method that attack question 2.

## 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 yeild intuitive results as it might match samples that do not correspond well. See figure 1.

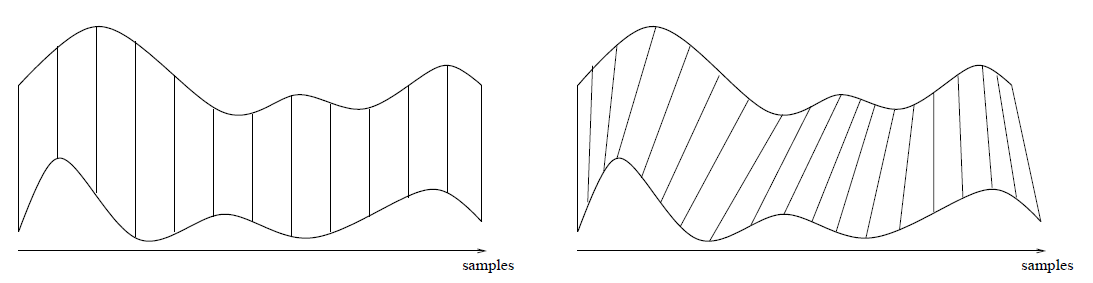


Figure 1 – The right scheme shows sample-by-sample naïve alignment after resampling and in the left scheme the alignment was performed using DTW

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. [4] [5] [6]

### 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.

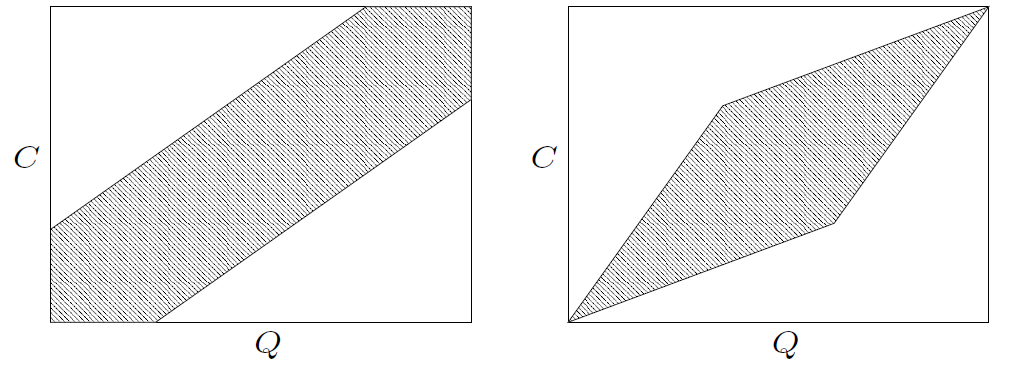


Figure 2 – Cost matrix constrains: Sukoe-Chuba Band (left) and the Itakura Parallelogram (right).

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. [6]

## 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. [7]

[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. [8] 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 [7] 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 [9] 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]

# Bibliography

|  |  |
| --- | --- |
| [1] | S. D. Connell, "ONLINE HANDWRITING RECOGNITION USING MULTIPLE PATTERN CLASS MODELS," Michigan, 2000. |
| [2] | R. Saabni and J. El-sana, "Efficient Generation of Comprehensive Database from Online Arabic Script Recognition". |
| [3] | F. Biadsy, R. Saabne and J. El-Sana, "Segmentation-Free Online Arabic Handwriting Recognition," 2010. |
| [4] | P. Senin, "Dynamic Time Warping Algorithm Review," Honolulu, USA, 2008. |
| [5] | T. M. Rath and M. Manmatha, "Word Image Matching Using Dynamic Time Warping". |
| [6] | S. Salvador and P. Chan, "FastDTW: Toward Accurate Dynamic Time Warping in Linear Time and Space," Melbourne. |
| [7] | I. Poitr and N. Thamper, "Fast Image Retrieval via Embeddings," Third International workshop on Statistical and Computional Theories of Vision, Nice, France, 2003. |
| [8] | "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]. |
| [9] | S. Shirdhonkar and D. W. Jacobs, "Approximate earth mover's distance in linear time," University of Maryland, Maryland, 2008. |