Recognition-based Segmentation of Online Handwritten Arabic Script

Or [Recognition-based Arabic Online Handwritten Word Segmentation]

George Kour

Faculty of Engineering

Tel Aviv University, Tel-Aviv Jaffa, Israel

kourgeorge@gmail.com

Raid Saabne

Faculty of Engineering

Tel Aviv University, Tel-Aviv Jaffa, Israel

Triangle R&D Center, Kafr Qarea, Israel

saabni@cs.bgu.ac.il

*Abstract*—the handwritten form of the Arabic language raises some challenges for automatic character segmentation and recognition. Character segmentation is a fundamental part in many Optical Character Recognition (OCR) systems. However, Correct and efficient segmentation of Arabic text into characters is considered to be an essential problem. Its importance is derived from the fact that incorrectly segmented characters are unlikely to be recognized correctly. The Arabic script is composed of strokes which contain a single or multiple connected letters. In this paper, we propose a novel online strokes-level recognition-based segmentation technique of Arabic script. The uniqueness and the main contribution of our approach is that most time consuming parts are done while the stroke is being scribed. The system consists of three stages. The first stage requires the most calculation effort thus is performed whilst the stroke is being written and contains two parts, the first is rules based engine to determine candidate segmentation points based on topological features; the second part is recognition-based scoring of the subsequences induced by the nominated segmentation points. The second stage is filtering of the topologically invalid segmentation points. And the last stage is an algorithm which determines the best subset of segmentation points. The system has been designed and tested using the ADAB Database. Promising results are obtained without using context help.

Keywords—Arabic Handwriting Recognition; Arabic Script Segmentation; Arabic Strokes Segmentation; Online Text Recognition

# Introduction

Handwriting remains the most used mean of communication and recording of information in the daily life. Therefore, a growing interest in the online character recognition field has taken place in the recent years.

Handwriting recognition (HWR) is a task of transforming a language represented in its spatial form of graphical marks into its symbolic representation. Handwriting recognition can be categorized into two main fields: online and offline recognition. Online handwriting recognition refers to the situation where the recognition is performed concurrently to the writing process. However, in the offline script recognition field, a digital image containing text is fed to the computers and the system attempt to recognize the written text [3- M. Al-Ammar, R. Al-Majed and H. Aboalsamh??]. The main existing approaches for script recognition are the holistic approach [11 - F. Biadsy, R. Saabni, and J. EL-SANA, ] and the analytic approach [add references]. 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. In the holistic approach, the recognition system needs to be trained over all words in the dictionary, while it is possible for small vocabulary of words; this is not feasible for large vocabularies (20,000 words or more). Since each word is constructed from a subset of the character alphabet, it is much more efficient to classify words using the analytic approach. [4]

The Arabic language is the fifth most used languages as a first language after Chinese, Hindi, Spanish and English. Around 350 million people use Arabic as their mother tongue. Most of them are citizens the Arab countries. Approximately 25 languages have adopted the Arabic alphabet slight some changes. Despite the fact that Arabic alphabets are used in many languages, the Arabic script recognition is at early stage in relation to the script recognition of Latin, Chinese and Kanji which has been a focus of study in the last decade and achieved an impressive recognition rates. The reason for this is mainly the difficulty in the nature of the Arabic script, lack of funds and other utilities such as text database, dictionaries, etc. [1 - Zeki, Ahmed M., Mohamad S. Zakaria and Choong-Yeun Liong]

The Arabic language is written right to left in a cursive manner in both handwritten and printed forms. Some Arabic letters has 2 main body shapes: Isolated and final, while other letters have 4 main body shapes: Isolated, Final, Initial and Medial. An occurrence of two shapes letters inside a word will lead to a split of the body into two or more parts, called Word-Parts (WP). Within the WP, the letters are connected in both handwritten and printed. Different letters may share the same body and only differ by additional strokes and dots called delayed strokes. In many cases Arabic word parts, which are connected when printed, are written in different strokes in handwritten script. A stroke may contain a single or multiple connected letters and may represent the main letter body or a delayed stroke we have mentioned earlier.

|  |  |
| --- | --- |
| C:\Users\kour\Desktop\1233141252420.tif  (a) | (b) |

1. The Tunisian city name kambot. As can be seen in (b) it contains 2 word parts (كمبو and ط). The main body of the first word part is written in 2 strokes (كـ and ـمبو ). The second WP contains only a single letter (ط) and is written using 2 strokes, the main body and an additional stroke.

|  |  |
| --- | --- |
|  |  |
| (a) | (b) |

1. Two letters groups which have the same main body and only differ by their additional strokes

In the analytic approach, character segmentation is a crucial part of the text recognition process. Correct segmentation of a word into letter is likely to result in a correct recognition. The recognition-based segmentation approaches the other way around is also valid, a good recognition system improves the segmentation precision. 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.

[talk about over and under segmentation in the general case, not specifically the Arabic]

Restricting our discussion to the analytic approach, the segmentation can be classified two main methods

* Dissection
* Recognition Based Segmentation

Dissection techniques learn the characteristic of the segmentation point and try to find these features in a candidate point. For example, in English cursive script segmentation a common feature is that segmentation point has local minima in the upper or lower contour of the word. Another feature is the slope of the segmentation point is low. Some techniques don’t try to segment a word to its letters but to graphemes, which are a combination of 2 or 3 letters or is a part of a letter. The recognition-based techniques operation is quite different. In principle no feature-based dissection algorithm is employed. Rather, the image is divided systematically into many overlapping pieces without regard to content. These are classified as part of an attempt to find a coherent segmentation / recognition result. The main interest of this category of methods is that they bypass the segmentation problem: no complex "dissection" algorithm has to be built and recognition errors are basically due to failures in classification. In recognition-based techniques, recognition can be performed by following either a serial or a parallel optimization scheme. In the first case recognition is done iteratively in a left-to-right scan of words, searching for a "satisfactory" recognition result. The parallel method proceeds in a more global way. It generates a lattice of all (or many) possible feature-to-letter combinations. The final decision is found by choosing an optimal path through the lattice [2]. A common approach that is followed by many researchers is over-segmentation of the text and validating each such candidate segmentation point by extracting feature vectors representing the segmented parts to some classifier or rules based engine. [12 - K. Daifallah, N. Zarka, and H. Jamous,11 - S. Abdulla, A. Al-Nassiri, and R. Salam]

In this paper we propose a novel approach which performs segmentation and recognition in the strokes level. We combine both holistic and analytic techniques for recognizing open dictionary Arabic online script. In section 2 we mention related Work done in the field of online Arabic recognition. In section 3, we describe the details of our approach. Results are displayed in the section 4. We discuss and conclude the work in section 5.

[Decorate the Introduction by citations]

# Related Work

A rules-based system for offline Arabic handwritten word segmentation was presented by Abdulla et al. in [11 - S. Abdulla, A. Al-Nassiri, and R. Salam]. Their method is based on extracting features of pixels lying on the upper contour. The freeman chain coding scheme was used to find the coordinates of the contour. The slope of the upper contour pixels is calculated and the direction of pairs of adjacent coordinate were marked by ‘+’ or ‘-‘. Segments were combined to formulate bigger decisive segments (DS). Set of possible segmentation point are nominated from the ‘+’ marked segments. These segmentation points were evaluated using a certain rules to find the final segmentation points (FSP). The system was tested on the demo version of IFN/INIT described in [10 - M. Pechwitz, S.] , and their own AHD/AUST database.

Randa et al. proposed a two stage word segmentation system of online Arabic handwritten text based on Hidden Markov Model (HMM) [4]. In the first stage, segmentation points were nominated by a simultaneous segmentation-recognition method using HMM. The proposed segmentation points were validated by a rules-based stage. Additional strokes were removed and not taken into consideration in both parts. The system was tested using a self-collected database (OHASD) that was described in [5].

Sari et al in their paper [6] proposed a method for offline Arabic Word parts segmentation, based on the topological characteristics of the word contour. They applied 8-conncted contour following algorithm to achieve a smoothed sequence of the X-Y coordinates of the outer contour. Then, identified Local minima in the lower outer contour to nominate Segmentation points and then formulated a rules based engine to identify valid segmentation points. The system was evaluated using a small database that contained 100 handwritten Arabic words sampled.

Laslo Digness et al. in [] proposed a segmentation based recognition approach for offline Arabic Handwritten words. Their method is based on dividing the word to smaller pieces which afterwards segmented into candidate letters and then classified into letter classes using statistical and structural features. Decisive tree were used to reduce the number of potential classes; neural networks to compute weights for all statistical features the output of this process was used as input for a k-NN classifier.

Khaled Daifallah et al. in [12 - K. Daifallah, N. Zarka, and H. Jamous] proposed a method for on-line Arabic handwritten words recognition. Their method operates on the stroke level. It established on segmentation-based recognition which contain several stages. The first stage proposes over-segmentation of the stroke. The segmentation points are selected by locating semi-horizontal lines moving from right to left. In a latter phase a portion of the segmentation points is filtered out by applying on a certain set of rules. Then HMM is used to classify the sub-strokes to letters using Hu feature. The candidate and its scoring letters are obtained. Based on tis results the set of best segmentation points are selected.

# Our Approach

Our approach employs both rules-based dissection and recognition-base segmentation techniques. The main idea of is to automatically segment a stroke while being scribed. A stroke is a subcomponent of a WP that spans from the pen down event to the corresponding pen up event. It may contain a single or multiple letters. We assume that each letter is contained entirely in a stroke, i.e. no letter span over multiple strokes. This assumption is valid for the majority of Arabic writing styles. The motivation […] (The only thing that needs to be really segmented is the stroke, and it is usually a standalone component, meaning that a reader can read what it contains) Thus being able to correctly segment and recognize the content of a stroke, cracks the Arabic script recognition problem. In this work our objective is to maximize the segmentation rate of Arabic handwritten words while maintaining low complexity and high performance of the system, rather than maximizing the recognition results.

The proposed technique can be combined with any letters recognition system and post processing system to achieve a complete word level online recognition technique. The proposed scheme defines a general scheme for ongoing Arabic script recognition based segmentation and can be used with different letter recognition techniques. In the figure below we describe a high level architecture of a complete Arabic Handwritten Word/Word-part segmentation-based recognition system employing our proposed recognition based segmentation approach.



1. Main System components

The approach consists of several stages.

**Stage 1 (Ongoing):** Ongoing Segmentation points nomination. Horizontal fragment (HF) identification. subsequences scoring calculation. Segmentation Scoring Matrix (SSM) construction. On every new Candidate point found, a new row and column are added to the matrix that contain the relevant subsequences scoring.

**Stage 2:** Redundant candidate points elimination and scoring correction.

**Stage 3:** Once the stroke is done (Pen Up Event). Segmentation points selection among the candidate points and. segmentation point selection algorithm is employed to select the most probable segmentation points.



High level system flow.

More details on every stage are provided in the subsections below.

## First Stage: Segmentation points nomination and substrokes scoring

***Preprocessing*:**

In the online handwriting recognition, the data is obtained by a digitizer. Most classification method requires the data to admit to several properties, such as size, vector sample length, etc… The data obtained by the digitizer may not have these properties, since it is affected but the handwriting speed, hand vibration and digitizer imperfection. The preprocessing is performed to overcome this mismatches and normalize the data to have a uniform structure. In order to avoid digitizer noises, we have performed preprocessing operations that include simplification using Douglas simplification algorithm and resampling to achieve a smooth equidistant sample points.

|  |  |
| --- | --- |
| (a) | (b) |

1. The letter د (D) before (a) and after(b) preprocessing.

***Horizontal Fragment identification:***

A stroke is represented by a sequence of points on the 2-dimensional space, 

|  |  |
| --- | --- |
|  | (1) |

Arabic segmentation points are usually contained in horizontal fragments (HF) of the stroke. In this stage such fragments are identified in the written text. Once an HF is detected, its medial point is set as a candidate point (CP). A HF is a subsequence of the stroke that is relatively straight, has a low slope and directed right to left. In the ongoing process to get the largest possible HF, we mark both the beginning and the end of the horizontal segment.

A pair of adjacent points is considered horizontal if the its slope is low relative the horizontal axis i.e. . This parameter was found empirically and independently on Daifallah, N. Zarka in [12 - K. Daifallah, N. Zarka]. Who has found the same parameter.

A horizontal segment may contain a two points at least. We try to identify the largest possible HF.



1. Horizontal Segments [HS] of the word جبل (JABAL)

Ideally the whole process should be initiated on every new point received by the digitizer, however, it is unnecessary since no much information is obtained by each new point, we invoke the process on every k point, when k is one of the Parameters of the system and set to be 5 in our system.

***Sub-strokes scoring:***

The candidate point  represents the index of the candidate point in the stroke. For the simplicity of the presentation we define  , i.e. the first point in the stroke , although the first point is not defined as a segmentation point. We also we define the last point inm the stroke as a candidate point, although it is not a candidate point but an obvious segmentation point. Both the “first candidate point” and the “last candidate point” will not be taken into account in the results sections.



1. Candidate points of the word لبيه. The first and last candidate points are colored in green. The red points are the actual candidate points.

A sub-stroke is a sub-sequence of the stroke  that starts at candidate point and ends at candidate point, i.e.

|  |  |
| --- | --- |
|  | (2) |

The sub-stroke  may represent a letter or a portion of a letter. In some cases it also may represent a multiple connected letters. The scores matrix contains the resemblance scoring for all sub-strokes.

|  |  |
| --- | --- |
|  | (3) |

is the classification scoring given by the letter classifier to the sub-stroke. It is s clear that is an upper triangular matrix. For the sake of good performance as well as segmentation correctness we add a locality constraint to improve performance and to avoid marginal segmentation. That is, we only calculate a narrow band of the  matrix above the diagonal. The band width is no larger than.

|  |  |
| --- | --- |
| if  or | (4) |

The matrixis calculated while the stroke is being scribed. Once a new candidate point is identified, a new row and column is added to the matrix with the corresponding for each added cell in the band of the lower triangular matrix. It can be easily noted that the number of cells calculates in the limited band iswhere  is the number of candidate points in the stroke.

A letters classifier that is described in depth in [5] is used to score the sub-strokes. For the sake of completeness, here we will outline the main idea of the classification system. The classifier uses Earth movers Distance embedding technique descried in [7] to project samples of the Arabic letters to a low dimensional space. Since the embedding is into the L1 space, given a sequence, the k-NN are found using k-dtree. The exact resemblance score is determined by a minimum score that is given by a predefined linear combination of the L1 distance in the embedding space and Sokoe-Chuba DTW.

The classifier receives a sequence and a position, and it contains 4 databases; each database contains letter samples in a certain position *(Ini, Mid, Fin or Iso)* regardless of the letter label. As mentioned before a stroke can contain a single or several letters. Here we enumerate all the letters positions options that a stroke can contain.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

Where represent more than one occurrence and  represents zero or more occurrences. The  matrix below demonstrates the databases in which the classifier will look into when trying to find the closest candidates. In Equation (3), the Greek letter represents the set of position databases that the sub-stroke may belong to:

|  |  |  |
| --- | --- | --- |
|  | (5) | |
|  | | (6) |

As mentioned before, for each subsequence, the recognition system returns a set of 3 potential letters candidates, with their resemblance scoring. In the current implementation we consider only the candidate with the best (minimal) scoring . In a future work we can evaluate other scoring technique that will be based on other scoring technique that will take into consideration all the relevant candidates.

In figure 5 and table [] we demonstrate the idea of the segmentation points nomination and the Subsequences matrix.

Table [] visually demonstrates the subsequences of the word shown in the figure above. Note that we have eliminated the first column and the last row of the matrix since it contains no subsequences.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  | N/A |  |  |  | N/A | N/A |
|  | N/A | N/A |  |  |  | N/A |
|  | N/A | N/A | N/A |  |  |  |
|  | N/A | N/A | N/A | N/A |  |  |
|  | N/A | N/A | N/A | N/A | N/A |  |
|  | N/A | N/A | N/A | N/A | N/A | N/A |

1. Visual demonstration of subsequences of the word shown in figure 7.

## Second Stage: Candidate points sieving and scoring correction

In this stage we eliminate redundant segmentation points. The elimination is based on the following rules:

**Rule 1**: Inner Segmentation point lies close to the baseline.

**Rule 2**: Segmentation points do not reside in loops.

**Rule 3**: A sub-stroke length/area is proportional to the stroke/length of the containing stroke.

Rule 1 is based on the fact that segmentation point lies on the baseline (see []), thus points that are majorly far from the baseline are eliminated. The baseline is proved to be an important piece of information in both segmentation and recognition domains of both online and offline handwriting recognition. In words recognition, it plays a role in determining if an additional stroke to differentiate between diacritic dots according to their position from the baseline (above or under).

Rule 2 is clear since in most writing styles, segmentation point do not reside inside a loop.

The third rule’s goal is to avoid high scoring resulted for discordant scaling of the letters; a scoring correction should be employed. It aims to reduce the effect of scaling problem. To illustrate the discordant scaling problem, see figure below. The suffix of the letter “d” is very similar to the letter “a” (Fin). The only way to visually discriminate between them is by comparing the scaling of this suffix to the whole stroke dimensions. Thus this phase refines the recognition results according the subsequence scale in accordance to the sequence scale. We have penalized sub-strokes that are disproportional to the stroke size by multiplying the score of each sub-stroke with a Gaussian normalized by the stroke area/Length.

In some cases candidate segmentation points are incorrectly nominated on areas that are not horizontal, this is caused from the fact that the nomination is done while the word is being scribed, and our filtering algorithm should be corrected to handle this case.

|  |
| --- |
|  |

1. Incorrect nomination of a candidate point that lays on a Segmentation fragmnet.

In several cases we encountered candidate points that reside on the same horizontal fragment. [why does it happen?]. We filter out this redundant candidate points in the nomination process and do not wait till this filtering phase. The reason is that since we calculate a narrow band of the Scoring matrix, leaving the impermissible candidate points will result in low performance since highly over segment the word will cause also the system to miss letters because we calculate only a portion a narrow band of the scoring matrix. If such case is identified by the nomination algorithm the medial point between the adjacent candidate points is taken.

## Third Stage: Segmentation Selection

The goal of this phase is to select the best segmentation points set among the candidate segmentation points. It is done by finding the best segmentation path in.

A path with the lengthin the scoring matrix  is defined as follows:

1. The first link: 
2.  for any 
3. 

Three algorithms are proposed in this work. The first we name as “Forward Segmentation Selection” (FSS) and the other is named the “Backward Segmentation Selection” (BSS) and the third named “Greedy Segmentation Selection” (GSS). The final segmentation is the segmentation that has the minimalbetween the FSS and BSS and GSS normalized by the number of Segmentation points. In the algorithm below,is the set of candidate points Including the pen-down point.

1. 
2. 
3. while 
   1. 
   2. 
   3. 
   4. 
4. End while
5. Forward Segmentation Selection (FSS)
6. 
7. 
8. while 
   1. 
   2. 
   3. 
   4. 
9. End while
10. Backward Segmentation Selection (FSS)

The idea in the following algorithm, is that in each iteration two segmentation point are selected, the first is the best score candidate point from the end of the stroke and the second is the best from the beginning. In this approach, it is less likely to step over segmentation point, like what happened in the FSS and BSS.

1. 
2. 
3. 
4. 
5. while 
   1. 
   2. 
   3. 
   4. Remove all cells that represent a candidate point that has an index lower than 
   5. 
   6. 
   7. 
   8. 
   9. Remove all cells that represent a candidate point that has an index larger than 
   10. 
6. End while
7. Forward Segmentation Selection (FSS)
8. 
9. 
10. 
11. while 
    1. 
    2. 
    3. 
    4. Remove all cells that represent a candidate point between  and.
12. End while
13. Greedy Segmentation Selection (GSS)

In GSS, in every iteration, the cell with the best scoring is selected thus the selected cell represents a sub-sequence that has the best resemblance for a letter, thus this subsequence must be included in the final segmentation. If cell  was selected, both the candidate point  and  is added to the final set and every segmentation point between  and  are removed from the scoring matrix.

# Validation

In ther works, usually a human taken place to make sure that the segmentation is correct. In this paper, we have used an automatic validation process. For each segmentation point found by our system, for it to be count as a true positive, the information measure between the real segmentation point (marked by an expert human) and the segmentation point sound by the system should be small.

## Subsequence Complexity measure

The aim of this measure is to give a numerical confidence that if two points reside on the relatively same straight line over the stroke. To demonstrate the idea see figure below. From another point of view, it can be thought about as the amount of information stored in a sub-sequence or the complexity of the sequence. This notion is used in 2 places in our work:

* The Horizontal fragment identification.
* In the segmentation point validation step.

To calculate the complexity measure of a sequence, first we simplify the subsequence. Then, for each angle  we calculate the parameter.

|  |  |
| --- | --- |
|  | (7) |

The complexity measure is the defined as:

|  |  |
| --- | --- |
|  | (8) |



1. A graphical representation of the angles of a sequece. It can be seen that per our defenition Sequence 1 Complexity measure is smaller than sequence 2 complexity measure.

# The database

The data is the most important part of any supervised learning technique. The data is used for both learning, validation and testing stages and has a critical effect on the system performance. In this work we have chosen to use the ADAB database. The ADAB database is de-facto a standard in the online Arabic handwriting recognition research field. It is freely available and consists of more than 20k Arabic handwritten words scribed by more than 170 different writers. The words are taken from the 937 Tunisian town/village names. It contains the trajectory information and a plot image of the word trajectory. The ADAB-database v.1 is divided to 3 sets. Details about the number of files, words, characters, and writers are detailed in [6]. Our training and testing set both are taken from this database.

The information in the ADAB database provides for each word its label and the strokes that were written by the writer, no information relating the strokes to letters or to word parts provided, thus, some work needed to be performed to add this information to the database. This edition is needed for providing letters samples for our classifier as well as for being able to automatically evaluate the segmentation rate. We have created a friendly UI system that reads the samples in the ADAB database, and a human professional segment the samples. The output of this process is an xml for each word sample that contain letter level information, and WPs level details. Since additional strokes are not our interest in the segmentation process, the system automatically filtered out additional strokes. The professional had the ability to filter our additional strokes that could not be identified by the system as such. We have manually segmented ~8k samples which consisted about ~20k strokes.

Our system reads an xml files, extracts the strokes and information we provided our training system with letters and also extracted strokes to test our system.

# Experimental Results

Our system was implemented in Matlab environment. The total number of samples in the test set is 200. As mentioned a sample is a Tunisian city name, a city name can contain 1 or more words.

Although our approach segments the written script in the stroke level, it is more reasonable to display the results in the WP and word levels. A correct segmentation of a word-part means that all the strokes of the in it were segmented correctly. A correctly segmented stroke is the case when all the letters in the stroke are correctly segmented. In table 1 below, you can see the length distribution of the Word parts in the test set. By length we mean the number of letters the word part contains.

The segmentation points results were validated automatically, if a word part was recognized correctly that means that the segmentation is correct. For each letter we select the 3 letter and shoe our recognition results based on if one of the candidate letters is the correct letter.

1. Segmentation Point Results

|  |  |
| --- | --- |
| Percision | 95.4% |
| Recall | 95.6% |
| Letters Recognition Rate | 87.27 % |

If the recognition was incorrect, each segmentation point recognized by the system was validated automatically by making sure that there is no much information between the segmentation ground truth and the segmentation provided by the system.

1. Word Part Results

|  |  |
| --- | --- |
| Total Number of Word Parts | 179 |
| Segmentation Rate | 87.7% |
| Recognition rate | 70.94% |
| Average time | 0.64332 [sec] |

1. Distribution of the WP Length

| Word-Part Length | Percentage of samples |
| --- | --- |
| 1 | 325 |
| 2 | 171 |
| 3 | 136 |
| 4 and more | 91 |

***Over-segmentation:***

To avoid a candidate point nomination in a horizontal fragment at the beginning of the stroke which is in most cases a false candidate point, we have added the limitation that segmentation point is nominated only if the sub-stroke that spans from the beginning of the stroke to the current point, is complex.

|  |  |
| --- | --- |
|  |  |

1. 2 samples of redundant Candidate point at the beginning of the firsr stroke.

|  |
| --- |
|  |

1. Over-segmentation in the letter س. These happens since this letter is very similar to the appearance to a combination of two consequtive ب (ـببـ) letters and only can be identified from the context and using the additional strokes

|  |
| --- |
|  |

1. Over-segmentation appeared frequently in the letter M, since it contains two horizontal segments (can be handled by eliminatinf candidate points in a loop).

***Under-segmentation:***

It may results from 2 reasons:

1. No point is get nominated in the horizontal fragment
2. Nominated but not selected.

The first mainly results from letter pairs that do not contain horizontal joint between them; this issue is partially solved by extending the notion of a letters to include such pairs of letters. For example the pair لم and لح, these letters may have 2-4 positions. The second may result from nomination of a candidate point on correct horizontal segment but in a late fraction of the segmentation fragment which result is a low scoring and thus not being selected by the Segmentation selection algorithm. This issue usually occurs in the letter و in its final position.

|  |
| --- |
|  |

1. An example of a late candidate point in the letter و.
2. Performance of each Segmentation selection algorithm

|  |  |  |
| --- | --- | --- |
| Segmentation Selection Algorithm | WP Segmentation rate | WP Recognition rate |
| FSS | 81.5% | 70.9% |
| BSS | 78.4% | 37.5% |
| GSS | 77.1% | 68.7% |
| FBSS | 83.34% | 73.743% |

Most error in the final segmentation is caused by segmentation selection algorithm. Thus we had to combine several segmentation selection approaches to achieve the best results.

We will show that the maximal number of letters does not affect much the segmentation rate and recognition rate, the graph will convergence after 200 samples per letter.

1. The diagram shows the influence of the number of letters samples on the segmentation (SR) , recognition rates (RR) , and the average segmentation time. All the three parameters showing convergance when the maximum letter samples per letter position is larger than 200.

# Future work

[we can fix the orientation problem in figure work by rotating by different angles but the ADAB sample are our system could easily handle samples to]

[Consider the delayed strokes segmentation: every strokes is classified to be in the main body or an additional stroke, id classified in the main body, then we handled it. If classified as a delayed stroke, it should be matched to the letter is related.

For every word, the key a serious of features such that

(A,Ham, KL): أكل

(D,Dot,HB): ذهب

##### References

1. Zeki, Ahmed M., Mohamad S. Zakaria and Choong-Yeun Liong. "Segmentation of Arabic Characters: A Comprehensive Survey." Technology Diffusion and Adoption: Global Complexity, Global Innovation. IGI Global, 2013. 251-288. Web. 16 Jun. 2013. doi:10.4018/978-1-4666-2791-8.ch016
2. R. G. Casey and E. Lecolinet, “A survey of methods and strategies in character segmentation,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 18, no. 7, pp. 690–706, Jul. 1996.
3. M. Al-Ammar, R. Al-Majed and H. Aboalsamh, "Online Handwriting Recognition for the Arabic Letter Set," Recent Researches in Communications and IT.
4. R. I. Elanwar, M. Rashwan, and S. Mashali, “Unconstrained Arabic Online Handwritten Words Segmentation using New HMM State Design,” pp. 1046–1054, 2012.
5. R. Elanwar, M. Rashwan, and S. Mashali, "OHASD: The first online Arabic sentence database handwritten on tablet PC", International Conference on Signal and Image Processing ICSIP 2010, Singapore, Proceedings of World Academy of Science, Engineering and Technology (WASET), vol. 72, pp.710-715, 2010.
6. S. D. Connell, “Online Handwriting Recognition Using Multiple Pattern Class Models,” *PH.D Thesis*.
7. My other paper
8. M. Kherallah, N. Tagougui, A. M. Alimi, H. El Abed, and V. Margner, “Online Arabic Handwriting Recognition Competition,” *2011 International Conference on Document Analysis and Recognition*, pp. 1454–1458, Sep. 2011.
9. S. Shirdhonkar and D. W. Jacobs, “Approximate earth mover’s distance in linear time,” *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, Jun. 2008.
10. M. Pechwitz, S. S. Maddouri, and V. Märgner, “IFN / ENIT - DATABASE OF,” pp. 1–8.
11. S. Abdulla, A. Al-Nassiri, and R. Salam, “Off-Line Arabic Handwritten Word Segmentation Using Rotational Invariant Segments Features,” *International Arab Journal of …*, 2008.
12. K. Daifallah, N. Zarka, and H. Jamous, “Recognition-Based Segmentation Algorithm for On-Line Arabic Handwriting,” *2009 10th International Conference on Document Analysis and Recognition*, pp. 886–890, 2009.
13. F. Biadsy, R. Saabni, and J. EL-SANA, “Segmentation-free online arabic handwriting recognition,” *… Journal of Pattern Recognition …*, pp. 1–24, 2011.