https://doi.org/10.17576/apjitm-2019-0801-08

http://www.ftsm.ukm.my/apjitm
Asia-Pacific Journal of Information Technology and Multimedia
Jurnal Teknologi Maklumat dan Multimedia Asia-Pasifik
Vol. 8 No. 1, June 2019: 97 – 127
e-ISSN: 2289-2192

ADVANCED STROKE LABELLING TECHNIQUE BASED ON DIRECTIONS FEATURES FOR ARABIC CHARACTER SEGMENTATION

TARIK ABDEL-KAREEM ABU-AIN
SITI NORUL HUDA SHEIKH ABDULLAH
KHAIRUDDIN OMAR
SITI ZAHARAH ABD RAHMAN

ABSTRACT

Offline Character segmentation of text images is an important step in many document image analysis and recognition (DIAR) applications. However, the character segmentation of both writing styles (printed and handwritten) remains an open problem. Moreover, the manual segmentation is time-consuming and impractical for large numbers of documents. Based on the unconstraint-cursive handwritten perspective, the auto character segmentation is more challenging and complex. The Arabic script writing style suffers from many common problems, such as sub-words overlapping, characters overlapping, and missed characters. These challenging issues have attracted the attention of researchers in the field of DIAR for Arabic character segmentation. The proposed method combines a new advanced Stroke Labelling based on Direction Features (SLDF2) technique and a modified vertical projection histogram (MVPH) technique. This technique extracts the relationship between each text stroke pixel and its 8 neighboring foreground pixels and labels it with the proper value before identify the possible segmentation points. The text preparation for the segmentation process was achieved using multiple preprocessing steps and developing an advanced stroke labelling technique based on direction features. Several Arabic language structural-rules were made to detect the candidate segmentation points (CSP), detect many character overlapping cases, solve the missed characters problem that appears as a result of using the text skeleton in VPH, and validate the CSP. All techniques and methods are tested on the ACDAR benchmark database. The validation method used to measure segmentation accuracy was a quantitative analysis that includes Recall, Precision, and F-measurement tests. The average accuracy of the proposed segmentation method was 92.44%, which outperforms the state-of-the-art method.

Keywords: Arabic character segmentation; document image analysis and recognition; candidate segmentation points; stroke labelling based on direction features; vertical projection histogram

INTRODUCTION

Text script is used as a daily communication channel between people and in archiving tasks. The script recognition process is considered to be easy for humans if they have prior knowledge of the script's composition structure. However, the same task becomes exhausting if a large number of documents need processing, which has encouraged many researchers to focus on this field with the goal of creating an automatic recognition system. Most of the recognition systems attempt to segment the text into its basic characters as a preprocessing step before proceeding to the feature extraction and recognition phases (Alginahi 2013; Parvez & Mahmoud 2013). Although an extensive amount of researches has been performed in the area of offline recognition of handwritten Latin scripts, a smaller number of works have been performed on Arabic scripts due to the difficulties associated with Arabic handwritten text segmentation (Alginahi 2013; Parvez & Mahmoud 2013, Firdaus, Khumaini & Utaminingrum, 2017, Ahmad et. al. 2017).

Segmentation becomes a vital issue for Arabic optical character recognition due to its cohesiveness, complexity and variety of writing styles. Similar to other languages, there are three Arabic writing styles; handwritten, printed and calligraphy. The Arabic calligraphy can be usually found in mosques, historic buildings and museums. It is often written by professional writers or calligraphers. Old Kufi, Kufi, Thuluth, Naskh, Roqaa, Diwani, Persian and Maghrebi (Andalusi) are the eight major types of Arabic calligraphy. The characteristics which differ across the calligraphy are special slanting, straightness, length and thickness (Bataineh, Abdullah, & Omar 2012). These character fashions entail novel segmentation techniques to explicitly overcome those unstructured challenges or difficulty. Up to date, Neural-based method introduced by Al Hamad and Zitar (2010) neglects such cases that characters and sub-words are overlapping. This aforementioned challenges endures from the over-segmentation problem. Figure 1(a) through (h) show examples of a sentence (al-khat lesan al-arab) with different structures or natures based on the type of calligraphy.

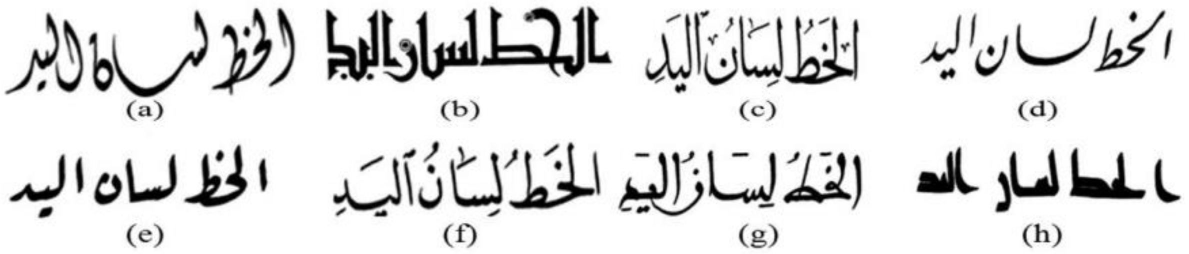


FIGURE 1. The (al-khat lesan al-yad) "the calligraphy is a tongue of hand" written in the main Arabic calligraphy types (a) Diwani, (b) Kufi, (c) Thuluth, (d) Persian, (e) Roqaa, (f) Naskh, (g) Andalusi, (h) Old Kufi (Bataineh et al. 2012).

In this work, the problem of segmenting the Arabic handwritten words into characters is addressed. This paper is organized as follows: the first section is the introduction and the second section presents the related works; the material and methods are illustrated in detail in section 3; the experimental results and discussion are discussed in section 4; and finally, the conclusions are presented in section 5.

RELATED WORKS

This section addresses the related works in Arabic handwritten text segmentation. Since the earlier works of Nazif (1975), Parhami and Taraghi (1981), and Amin, Masini, and Haton (1984), several methods have been proposed for the purpose of segmenting Arabic words into characters. Various researchers have used Arabic handwritten text segmentation (Al Hamad & Abu Zitar 2010; Ghaleb, Nagabhushan, & Pal 2017), skeletons of words (Al Hamad & Abu Zitar 2010; Wshah, Shi, & Govindaraju 2009), traces of the contours of words (Abdulla, Al-Nassiri, & Salam 2008; Wshah et al. 2009), template matching (Nidal Lamghari FSTG, Charaf, & Said Raghay FSTG 2016), neural networks (Al Hamad & Abu Zitar 2010; Ramdan, Omar, & Faidzul 2017), line adjacency graphs (Zidouri, 2006), morphological operations (Al-Badr & Haralick 1995), and recognition-based methods (characters are segmented while being recognized) (Husam A. Al Hamad & Abu Zitar 2010; Elnagar & Bentrcia 2015) to perform the segmentation process. Many reviews and surveys of both Arabic handwritten and printed text segmentation and recognition have been published (Ahmed et al. 2019; Radhiah et al. 2018; Al-Helali & Mahmoud 2017; Alginahi 2013; Parvez & Mahmoud 2013).

Abdulla et al. (2008) proposed a segmentation method for Arabic handwritten text by calculating the slope between the adjacent pixels of the text's upper contour and extracting the rotational invariant segment features to detect the nominate segment points. Afterward, a set of special threshold values and rules are applied to choose the final segment points. However, the algorithm suffers from issues such as under-segmentation and over-segmentation. In 2009, an algorithm for segmenting Arabic words into smaller segments was presented by Wshah et al. (2009). It relies on finding the segment path and detects the segment points depending on the exterior contour, skeleton and intersection points, which may contain up to three letters; also, one character may be segmented into five pieces. However, the method did not aim to segment the text into its exact characters. Alaei, Nagabhushan, and Pal (2010) proposed another segmentation method based on vertical projection histogram technique after finding the baseline of the text and straightening it. To validate the segmentation points, a set of baselines and many dependent rules are used. However, the algorithm suffers from all the types of overlapping, and it depends on baseline information (stroke width), where failure to find the accurate baseline leads to failure of the segmentation process.

An Arabic handwritten text segmentation technique based on the text skeleton was proposed by Al Hamad and Abu Zitar (2010). A neural based process is used to validate the segmentation points using a set of directional features from the text contour. However, the system is unable to detect characters and sub-words overlapping, and it suffers from the over-segmentation problem. Lawgali, Bouridane, Angelova, and Ghassemlooy (2011) proposed an algorithm for Arabic text segmentation based on both baseline detection and vertical projection. A set of rules are used to validate the segmentation points, such as branches points locations, start points locations, end points locations and a threshold value between the segmentation points and baseline. However, the algorithm depends on the baseline, which causes the segmentation process to be affected by the accuracy of baseline detection, and it is thus a less invariant approach. In addition, it is incapable of dealing with all the overlap cases, and it suffers from over-segmentation.

A multi-agent system to segment Arabic handwritten text was proposed by Elnagar and Bentrcia (2012). It detects the baseline as well as a set of featured points, such as start, end and branch points. Afterwards, multiple steps (agents) are employed to detect the points that should not contain any segmentation points by tracing the writing path of the thinned text. The final segmentation points are assigned into the middle of the remaining parts of the previous filtering process. However, their method cannot handle the overlapping problem. In addition, it cannot address certain cases, such as س,ص, ـدand the case of multiple successive uses of end points that look like ـسـ; it also only considers one type of style, "Naskh".

Later, Elaiwat and Abu-Zanona (2012) proposed a better segmentation model based on detecting the branch points on the baseline and applying a set of threshold values to find the final segmentation points. However, the model cannot detect the segmentation points in character overlapping cases, and it depends on the baseline detection accuracy and a set of threshold values, which makes it very sensitive to writing style. In 2012, Eraqi and Abdelazeem (2012) presented a technique to segment Arabic handwritten text into its basic graphemes. It applies the Douglas-Peuker algorithm on the thinned text to obtain linear curves. A set of direction features and geometric information of the Arabic text's nature are used to detect the segmentation points. However, the technique suffers from the over-segmentation problem and cannot handle the overlapping character problem.

ARABIC SCRIPT SEGMENTATION ISSUES

Arabic script is spoken by and have influenced on millions of people and a vast array of civilizations all around the world (Gordon & Grimes 2005; Ali and Suresha, 2019). Arabic script is a cursive font style. The characters are connected together with ligatures. A ligature is defined as a horizontal stroke which is used to connect two characters and contains the segmentation point. Many DIAR applications address the separated characters instead of the complete text/word.

In total the Arabic script consists of 28 characters, and each character takes from two to four different shapes, depending on its position in the word (Table 1). Some unsolved issues related to the field of Arabic script segmentation and recognition (Alginahi 2013; Parvez & Mahmoud 2013) due to unique nature of Arabic script compared to Latin scripts, such as:

1. It is always cursive, where each character can be joined with another character.
2. Characters can have two to four shapes according to their position in the word, which makes the number of basic character shapes 100 instead of 28. (Table 1).
3. Characters can be written in a different shape in spite of having the same position in the word (Figure 1-a).
4. The unconstraint of handwriting leads to the non-alignment of text components in either vertical or horizontal space, text slope, slant and skew problems. (Figure 1-b).
5. Certain characters use supportive objects called dots and diacritics (Figure 1-c).
6. Two types of overlapping exist: sub-words overlapping and characters overlapping (Figure 1-c).

For all of these reasons, the character segmentation phase is considered a crucial step in any Arabic character recognition and in many document analysis applications that extract features at the character level, such as writer identification (Chahi, El khadiri, El merabet, Ruichek, & Touahni 2018). In addition, segmentation has a considerable role in reducing the complexity of recognition systems because the number of target (recognition) classes will be limited to the number of Arabic letters only, instead of all possible Arabic words. On the contrary, recognition systems that depend on the lexicon may contain many hundreds of thousands of target classes for possible words that can be formed by the combination of single letters (Alginahi 2013; Parvez & Mahmoud 2013).

TABLE 1. The Arabic Script: 28 primary characters and their shapes

Beginning	Middle	End	Isolated
-	-	ـــأ	أ
بـــ	ـــبـــ	ـــب	ب
تـــ	ـــتـــ	ـــت	ت
ثـــ	ـــثـــ	ـــث	ث
جـــ	ـــجـــ	ـــج	ج
حـــ	ـــحـــ	ـــح	ح
خـــ	ـــخـــ	ـــخ	خ
-	-	ـــد	د
-	-	ـــذ	ذ
-	-	ـــر	ر
-	-	ـــز	ز
ســـ	ـــســـ	ـــس	س
شـــ	ـــشـــ	ـــش	ش
صـــ	ـــصـــ	ـــص	ص
ضـــ	ـــضـــ	ـــض	ض
طـــ	ـــطـــ	ـــط	ط
ظـــ	ـــظـــ	ـــظ	ظ
عـــ	ـــعـــ	ـــع	ع
غـــ	ـــغـــ	ـــغ	غ
فـــ	ـــفـــ	ـــف	ف
قـــ	ـــقـــ	ـــق	ق
كـــ	ـــكـــ	ـــك	ك
لـــ	ـــلـــ	ـــل	ل
مـــ	ـــمـــ	ـــم	م
نـــ	ـــنـــ	ـــن	ن
هـــ	ـــهـــ	ـــه	ه
-	-	ـــو	و
يـــ	ـــيـــ	ـــي	ي

MATERIAL AND METHODS

In this section, our proposed Arabic character segmentation system is described in detail. We proposed an Arabic Character Segmentation system using advanced stroke labelling based on direction features (ACS-SLDF2). The method over-segments Arabic words/sub-words into characters/sub-characters that may be processed further based on proposed Arabic language structural rules to achieve the best character segmentation accuracy. In summary, an overview of the various components that perform the proposed segmentation method is given in Figure 2.

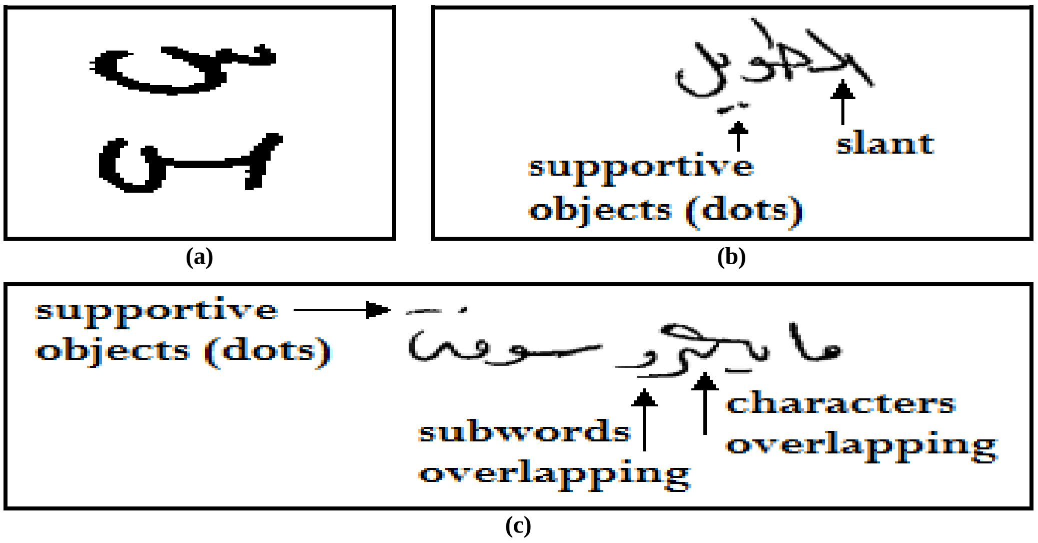
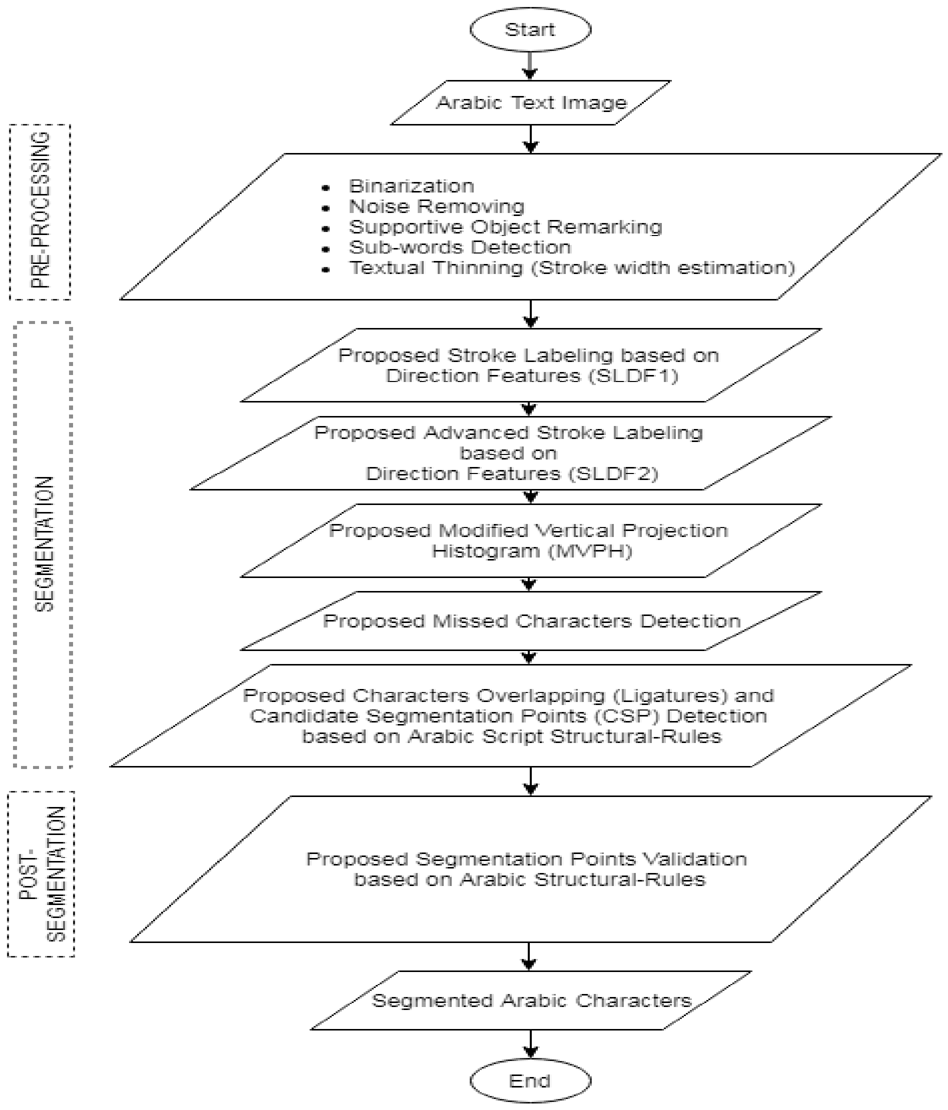


FIGURE 2. Arabic Script Issues, (a) different shape of the same character س, (b) text slant problem and supportive objects existence, (c) overlapping problem

Figure 3 shows the components of segmentation method involved and the list below are the terminology definitions used in the Figure 3.

1. **Horizontal projection histogram (HPH)**: Each row becomes a bin in the histogram. The count that is stored in a bin is the number of 1-pixels that appear in that row.
2. **Vertical projection histogram (VPH)**: Each column becomes a bin in the histogram. The count that is stored in a bin is the number of 1-pixels that appear in that column.
3. **Candidate segmentation point (CSP):** The point that is located using a segmentation method to disjoint the cursive text to its preliminary characters, which will become an actual segmentation point after validation rules.
4. **Actual segmentation point (ASP):** The candidate segmentation point that is verified using a set of rules that represent the true segmentation point.



Start
Arabic Text Image
PRE-PROCESSING
Binarization
Noise Removing
Supportive Object Remarking
Sub-words Detection
Textual Thinning (Stroke width estimation)
SEGMENTATION
Proposed Stroke Labeling based on Direction Features (SLDF1)
Proposed Advanced Stroke Labeling based on Direction Features (SLDF2)
Proposed Modified Vertical Projection Histogram (MVPH)
Proposed Missed Characters Detection
Proposed Characters Overlapping (Ligatures) and Candidate Segmentation Points (CSP) Detection based on Arabic Script Structural-Rules
POST-SEGMENTATION
Proposed Segmentation Points Validation based on Arabic Structural-Rules
Segmented Arabic Characters
End

FIGURE 3. The components of the proposed Arabic character segmentation method

Each component will be discussed further from pre-processing step to post-segmentation step.

PRE-PROCESSING TECHNIQUE

This research addresses offline document images that could affect during the acquisitioning process. Many preprocessing techniques are needed to improve the results of the succeeding steps which are binarization, noise removing, supportive object remarking, sub-words detection and textual thinning.

The binary format (Binarization) and noise removing techniques are employed to simplify the process of dealing with the document image contents (foreground "text" and background pixels). The binarization method from the work of Bataineh, Abdullah, and Omar (2011) is used to convert the gray scale image into a binary image ("1" indicates a foreground pixel, "0" a background pixel). In addition, a salt and pepper filter is used to remove the noise in the text. An example is given in Figure 4(b).

The proposed segmentation method uses the vertical projection histogram to locate the possible segmentation points. Because the segmentation points exist only in the main text body, it is necessary to detect and neglect the supportive objects (dots and diacritics) in segmentation processes by using dots (supportive objects) remarking technique. Most properly, the density of any dot is smaller than that of any other isolated character. These dots are remarked (relabeled) with the new value “9” instead of “1” because they will be used in the post-segmentation (validation) stage later. This technique fails for some handwritten documents, where some isolated characters, such as رor د, are written in a small size, causing them to be recognized as supportive objects. This disadvantage may be overcome by applying the algorithm in the upper and lower regions of the word\line only. These regions are detected by applying the horizontal projection histogram (HPH) using Equation 1, as illustrated in Figure 4(c). More details on supportive objects exclusion can be found in the work of Abdullah, Al-Harigy, and Al-Fraidi (2012).

$$HorizontalProjectionHistogram\ (x) = \textstyle\sum_y P(x,y). \qquad (1)$$

Furthermore, the process of uniformization of the text width is a crucial step in this research because it helps in locating the CSPs more accurately, as will be explained in the next section. Additionally, it speeds up the entire algorithm execution time by minimizing the number of image text pixels. A robust one pixel width thinning algorithm named as textual thinning (stroke width estimation) technique is used, which keeps the topology and connectivity of the text’s shape with the minimum number of foreground pixels (Abu-Ain, Abdullah, Bataineh, & Omar 2013), as Figure 4(d).

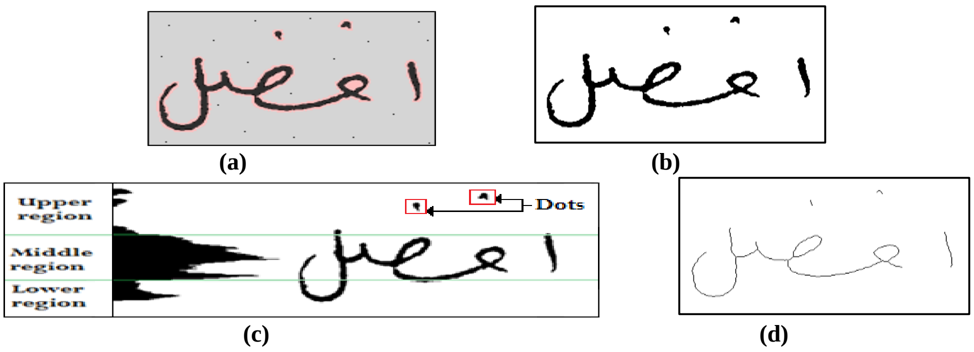


FIGURE 4. Preprocessing stage results (a) Original text image (b) binarization and noise removal (c) regions detection, and (d) skeleton/stroke extraction

PROPOSED ARABIC CHARACTER SEGMENTATION METHOD

In this research, a novel Arabic character segmentation is proposed based on advanced strokes labelling using direction features (SLDF2) and modified vertical projection histogram (MVPH) based on Arabic script writing structural-rules. The SLDF2 technique extracts the relationship between each text stroke pixel and its 8 neighboring foreground pixels and labels it with the proper value (refer to sub-sections A and B below). Afterward, a vertical projection histogram is calculated for the labeled text skeleton to detect the possible segmentation points.

A. *Text Stroke Labelling Based On Direction Features (SLDF)*

The first step is assigning the text stroke direction features. After representing the text image in the form of skeleton and binary discounting noise and dot, it defines the direction values, which are as follows: “2” for the vertical direction, “3” for the right diagonal direction, “4” for the horizontal direction, and “5” for the left diagonal direction (shown in Figure 5).

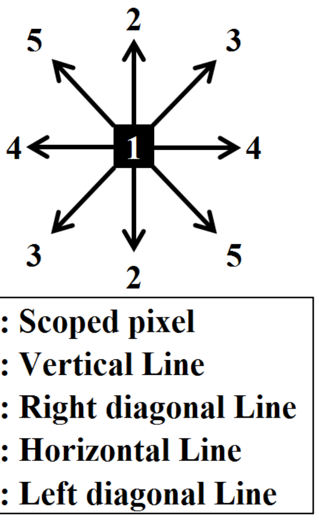


FIGURE 5. Direction values used for stroke labelling based on direction features (SLDF)

Apart from Al Hamad and Abu Zitar (2010) attempt to represent text images stroke via labelling the pattern based on direction features, this research proposes a new technique to assign and normalize direction of the text strokes. The overall processing time is minimized, where the text image is scanned one time only to assign the direction feature labels for all pixels in the image, instead of 4 scan iterations as proposed by Husam A. Al Hamad and Abu Zitar (2010). Figures 6 and 7 illustrate the flow and steps of the new technique to locate and normalize directions of each character image.

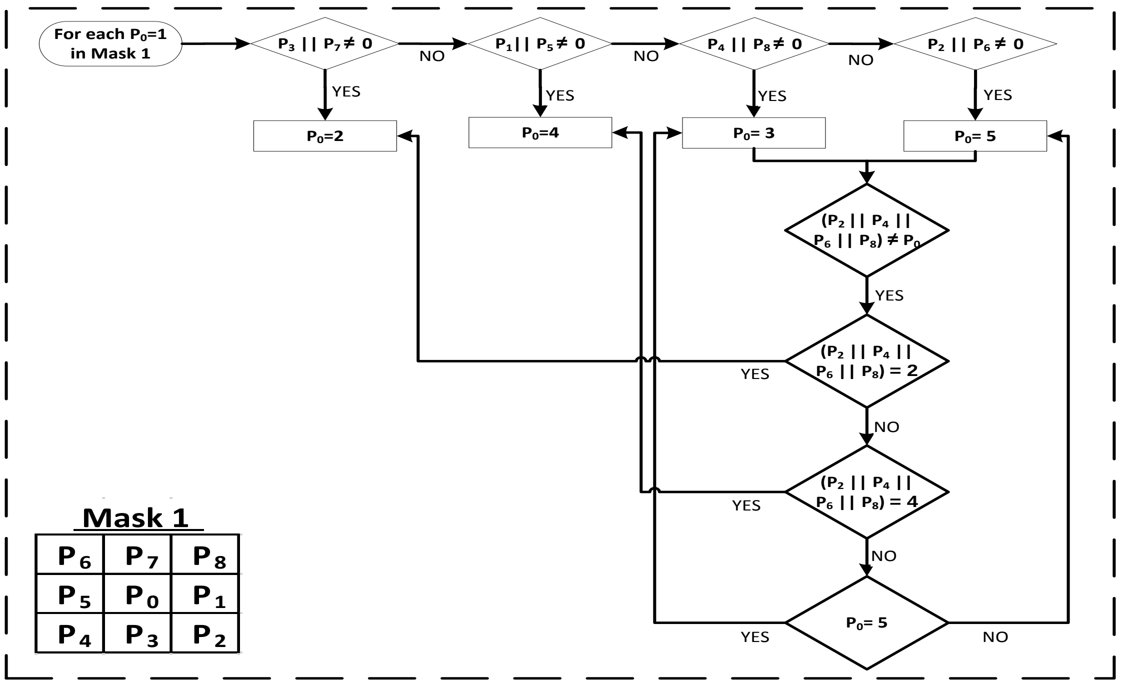


FIGURE 6. The proposed flowchart of stroke segment labelling based on direction features (SLDF1)

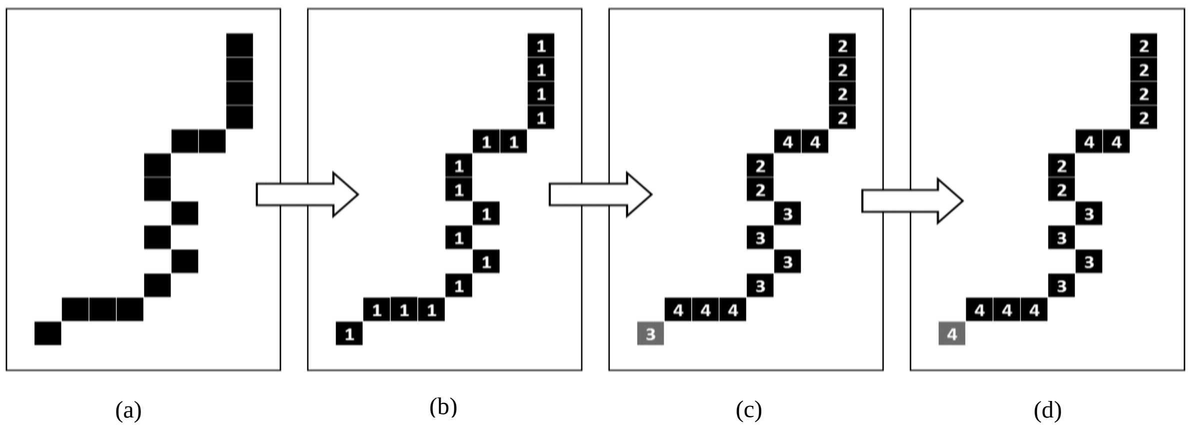


FIGURE 7. Steps of the new technique (SLDF1) to locate and normalize directions: (a) original stroke, (b) stroke in binary format, (c) distinguish directions, and (d) normalize the direction (gray shaded pixels)

In reference to the basic SLDF directions as shown in Figure 5, the following masks in Figure 8 are proposed to represent the text skeleton by assigning the direction values as follows:

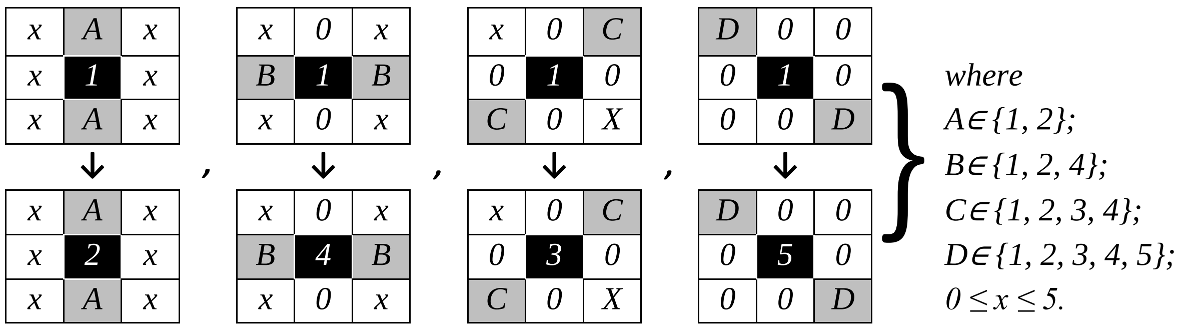
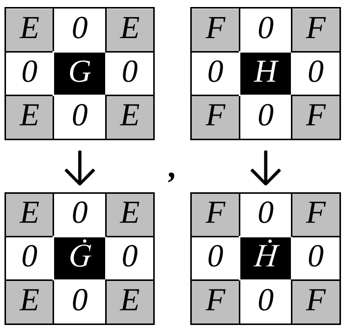


FIGURE 8. Masks represent the text skeleton

The following pseudo code represents the previous masks:

```
For each foreground pixel $P_0$= 1, do the following (refer to “Mask 1” in Figure 6):
    If ($P_3$ || $P_7$) $\neq$ 0 then $P_0$= 2
      Else if ($P_1$ || $P_5$) $\neq$ 0 then $P_0$= 4
        Else if ($P_4$ || $P_8$) $\neq$ 0 then $P_0$= 3
          Else if ($P_2$ || $P_6$) $\neq$ 0 then $P_0$= 5.
```

There are three steps to normalize the labelled strokes. By taking into account Figure 5’s representation, those steps are as follows: firstly, it finds spurious labeled pixels that represent less than two successive pixels (occurring in either right diagonal or left diagonal directions only), secondly it identifies its 8 neighboring labeled pixel values, and finally, it replaces the spurious pixel label by the proper value as illustrated in the following Figure 9:



where
$E \in \{0, 2, 4, 5\};$
$F \in \{0, 2, 3, 4\};$
$G=3;$
$H=5;$
$\dot{G}$ = {*if any* $E = 2$ *then* $\dot{G} = 2$, *else if any* $E=4$ *then* $\dot{G} = 4$, *else if any* $E=5$ *then* $\dot{G} = 5$};
$\dot{H}$ = {*if any* $F = 2$ *then* $\dot{H} = 2$, *else if any* $F=4$ *then* $\dot{H} = 4$, *else if any* $F=3$ *then* $\dot{H} = 3$}.

FIGURE 9. Labelled strokes normalization

The following pseudo code represents the prior masks:

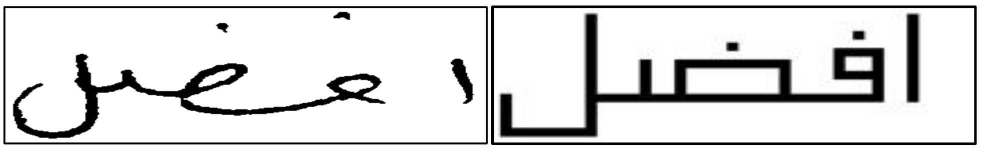
```
Raster scans the entire text image from any corner
   If ($P_0$ and $P_j$) = 3 or ($P_0$ and $P_j$) = 5 then do nothing
       Else if $P_0$ = (3 or 5) and $P_j$ = 2 then $P_0$ = 2
           Else if $P_0$ = (3 or 5) and $P_j$=4 then $P_0$ = 4
```

where, $j \in \{2, 4, 6, 8\}$, P_j is the 4 neighbours of target pixel P_0 ("Mask 1" in Figure 6).

The proposed technique performs the normalization in one step only, unlike the Al Hamad and Zitar's technique that takes two steps.

B. Advanced Stroke Labelling Based On Direction Features (SLDF2)

The direction features are used in the recognition of the text segment characters in Blumenstein et al.'s technique, and as a part of the validation of the text segmentation system in Al Hamad and Zitar's technique. However, we observe that these direction features could be adapted, with proper modifications, for use in the main part of the text's character segmentation, validation and detection of ligatures (overlapping of characters). In this research, a novel technique for advanced text stroke labelling based on direction features (SLDF2) is proposed. The number of extracted labels from the previously proposed SLDF was reduced from four labels to only two labels. In addition, the closed shape (hole) is the main component of the Arabic text structure, where 14 out of 28 letters have at least one hole such as in Figure 10. Without these modifications, the number of rules needed to perform the segmentation, characters overlapping and validation processes are inconceivable.



(a) (b)

FIGURE 10. (a) and (b) the example of letters with hole

For that reason, the proposed Advanced SLDF2 is inspired by the fact that alphanumeric text can be represented in digital (angled) form, which consists of only horizontal and vertical strokes (Figure 11).

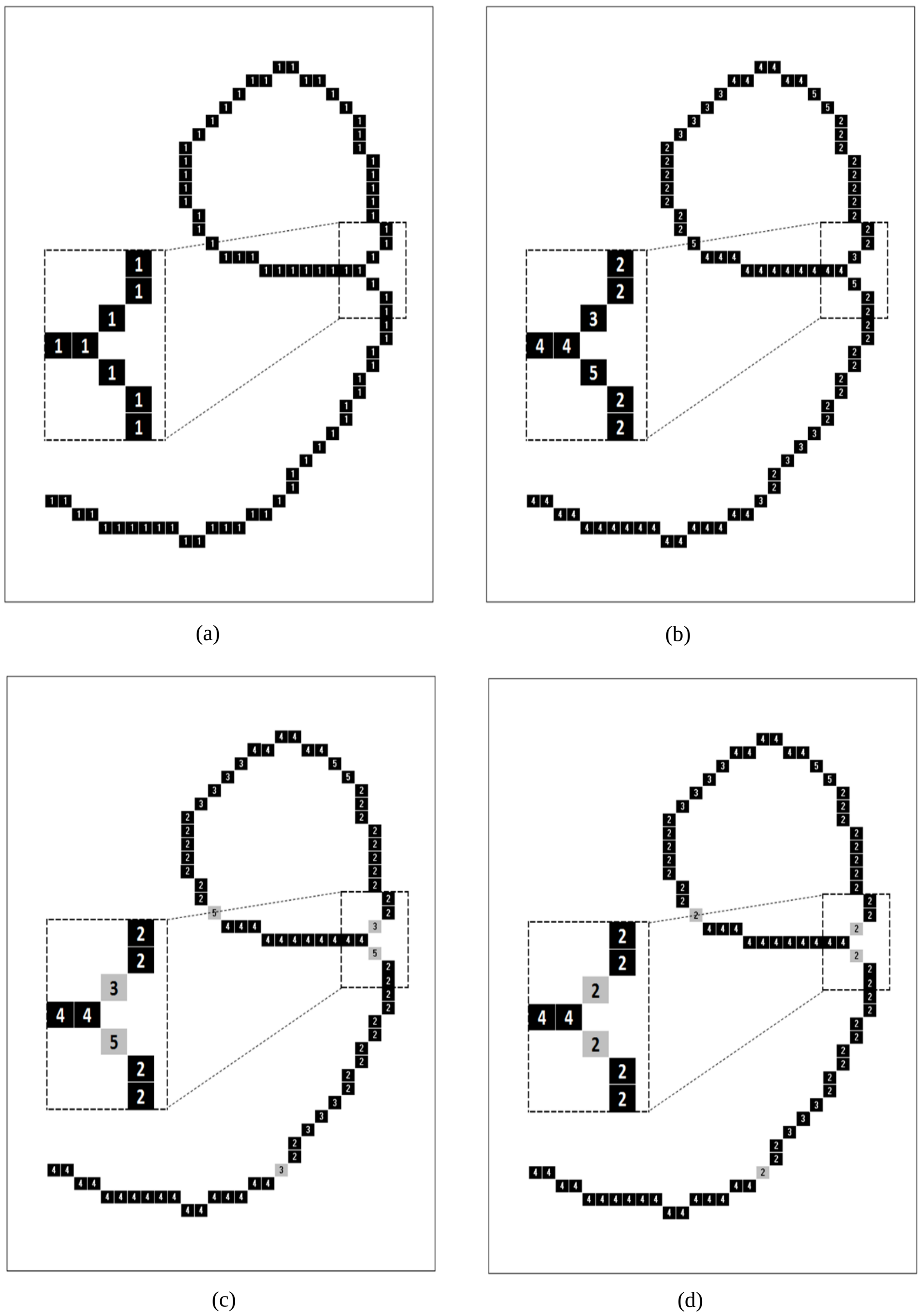


FIGURE 11. Alphanumeric text (a-b) handwritten format, and (c-d) digital angled format

The following stages are proposed to prepare the text pattern and assign the direction labels:
Stage 1: Direction labels reduction
Here, we reduce the number of direction values to two instead of four. Referring to the direction representation in Figure 5, the following steps are proposed to reduce the number of direction labels of the text pattern as Figure 12 follows:

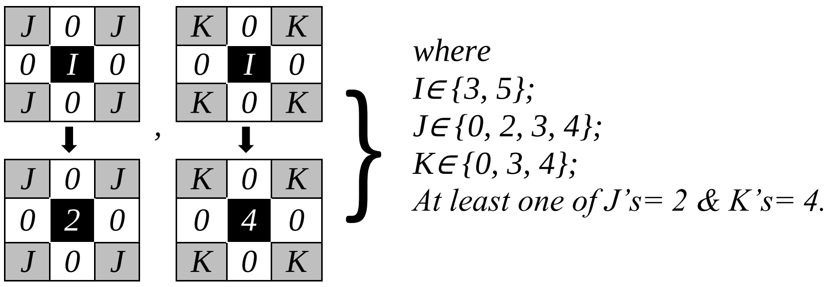


FIGURE 12. The number of direction labels of the text pattern

The following pseudo code represents the earlier masks:
Raster scans the entire text image row by row two times (one from the top left corner and one from the bottom right corner) separately using mask 1 (Figure 13(a)):

```
If $P_0$ = (3 or 4) then
   If any of $P_i$ = 2 then $P_0$ =2
      Else if any of $P_i$ = 4 then $P_0$ =4;
where i ∈{2, 4, 6, 8}.
```

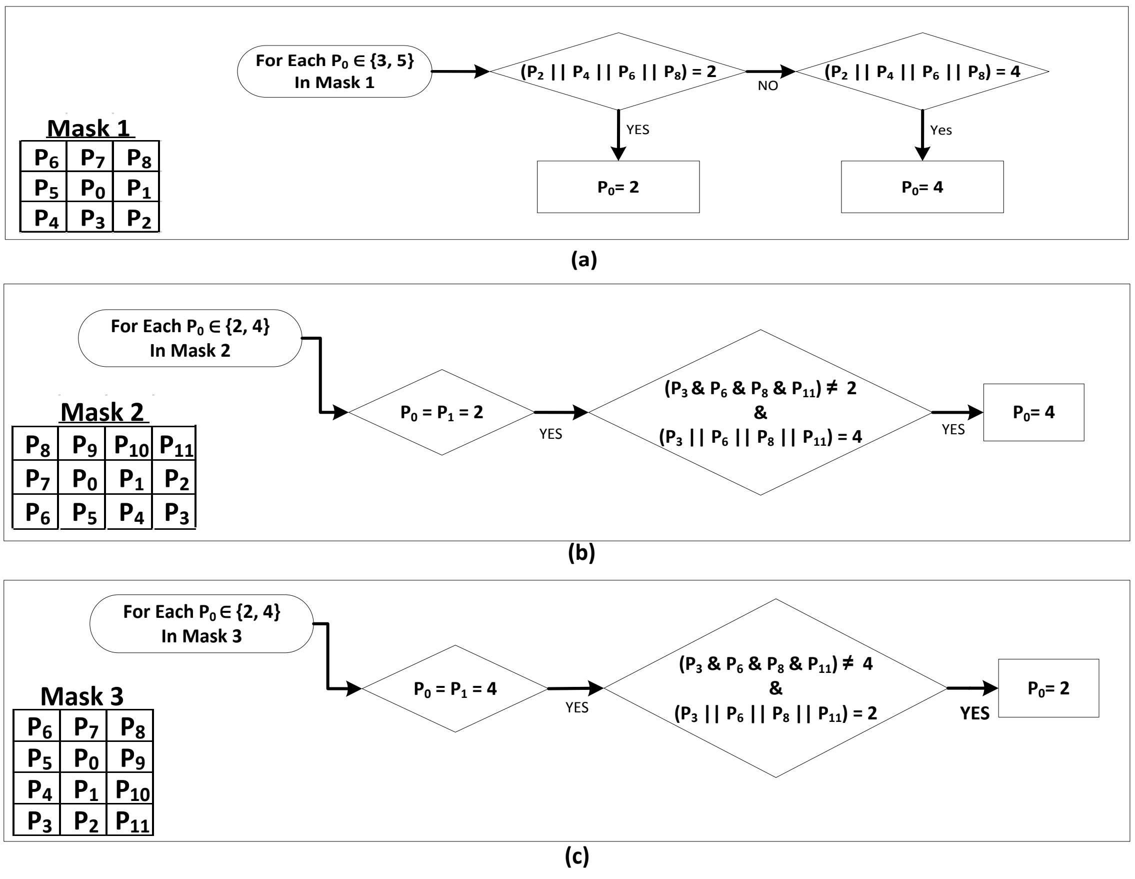

For Each P0 ∈ {3, 5} In Mask 1
(P2 || P4 || P6 || P8) = 2
NO
(P2 || P4 || P6 || P8) = 4
YES
Yes
P0= 2
P0= 4
Mask 1
P6 P7 P8
P5 P0 P1
P4 P3 P2
(a)
For Each P0 ∈ {2, 4} In Mask 2
Mask 2
P8 P9 P10 P11
P7 P0 P1 P2
P6 P5 P4 P3
P0 = P1 = 2
YES
(P3 & P6 & P8 & P11) ≠ 2 & (P3 || P6 || P8 || P11) = 4
YES
P0= 4
(b)
For Each P0 ∈ {2, 4} In Mask 3
Mask 3
P6 P7 P8
P5 P0 P9
P4 P1 P10
P3 P2 P11
P0 = P1 = 4
YES
(P3 & P6 & P8 & P11) ≠ 4 & (P3 || P6 || P8 || P11) = 2
YES
P0= 2
(c)

FIGURE 13. The flowchart of the novel technique of advanced stroke labelling based on direction features (SLDF2) (a) direction labels reduction, (b) 1st step of horizontal stroke segment normalization, (c) 2nd step of vertical stroke segment normalization

Stage 2: Spurious pixel normalization
Spurious pixels represent less than three successive pixels. From experiments, these pixels affect the process of detecting the segmentation point as well as the validation process. The following steps are proposed to detect and relabel the spurious pixels as shown in Figure 14:

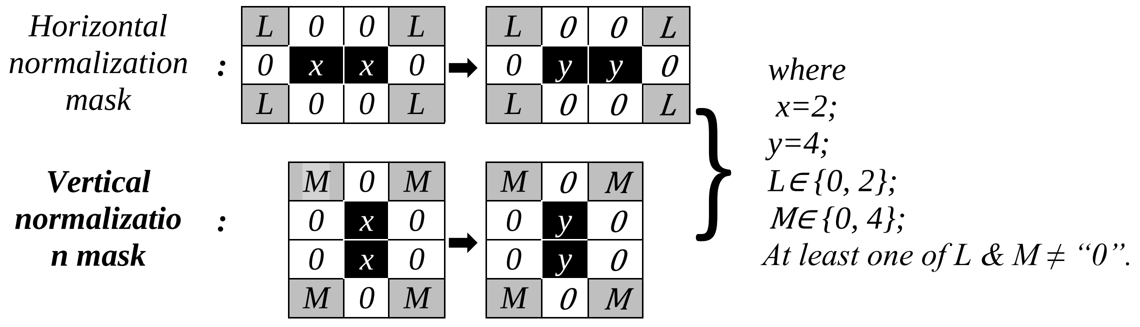

Horizontal normalization mask :
L 0 0 L
0 x x 0
L 0 0 L
L 0 0 L
0 y y 0
L 0 0 L
Vertical normalization mask :
M 0 M
0 x 0
0 x 0
M 0 M
M 0 M
0 y 0
0 y 0
M 0 M
where x=2; y=4; L∈{0, 2}; M∈{0, 4}; At least one of L & M ≠ "0".

FIGURE 14. The detection and relabeled the spurious pixels.

The following pseudo code represents the prior masks:

1. *Horizontal normalization:* Raster scan the entire text image using mask 2 (Figure 13(b)): If (P_0 & P_1 =4) and (P_3 & P_6 &P_8 & P_{11}≠4) and (P_3 || P_6 || P_8 || P_{11} =2) then (P_0 & P_1 =2), where the size of mask 2 is 4×3 to detect the two spurious pixels that are adjacent horizontally.
2. *Vertical normalization:* Raster scan the entire text image using mask 3 (Figure 13(c)): If (P_0 & P_1 =2) and (P_3 & P_6 & P_8 &P_{11}≠ 2) and (P_3 || P_6 || P_8 || P_{11}=4) then (P_0 & P_1 =4), where the size of mask 2 is 3×4 to detect the two spurious pixels that are adjacent vertically.

Stage 3: Closed shape (hole) detection:
Depending on Arabic script character structure, more than 50% of the characters in all possible shapes (beginning, middle, end and isolated) contain a close shape (hole). These holes play a significant role in our system because they are useful for avoiding the segmentation points inside them, and they are also used in validation of the segmentation points and detection of the ligatures (character overlapping).

In this research, we apply the connected component labelling technique (Stockman & Shapiro 2001) on the background pixels to detect the holes. The foreground pixels that surrounded these holes will be assigned as label "6". The entire process for determining text pixels' direction and normalization is illustrated in Figure 15 below:

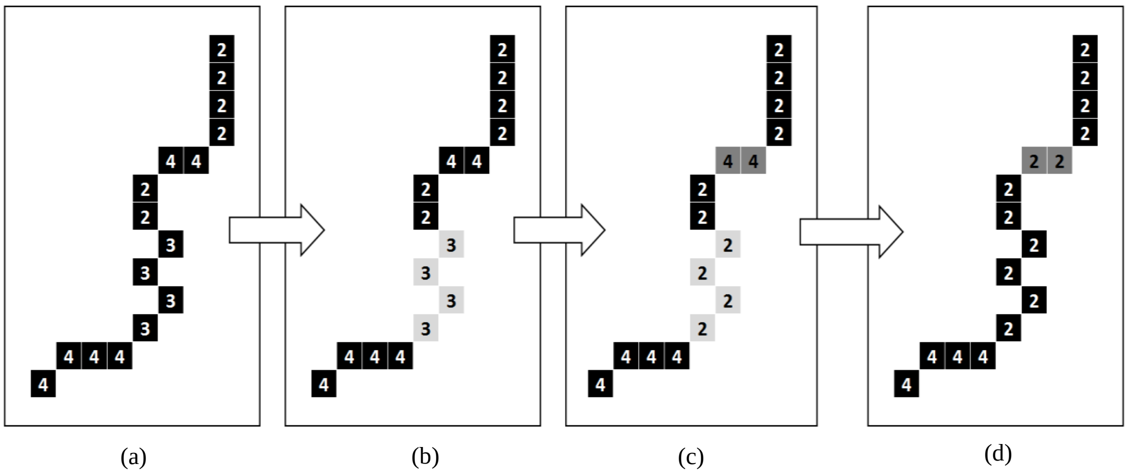


FIGURE 15. Steps of the new advanced technique (SLDF2) to locate and normalize directions: (a) original stroke, (b) stroke in SLDF1 format (light gray shaded pixels that need relabeled), (c) stroke in SLDF2 format and (d) normalizing the direction (dark gray shaded pixels)

The following pseudo code represents the prior closed shape detection process:

```
1.  On the first iteration:
    1.1. Raster scans each pixel of the text image,
    1.2. If the pixel is background,
         1.2.1.  Check the 8 neighbouring pixels of the current background pixel,
         1.2.2.  If there are no background neighbours, uniquely label the current element,
         1.2.3.  Otherwise, assign it to the smallest label of the background neighbour,
         1.2.4.  Store the correspondence between neighbouring labels.
2.  On the second iteration:
    2.1. Raster scans each pixel of the text image,
    2.2. If the pixel is background,
         2.2.1.  Relabel the background pixel with the lowest correspondence label.
3.  On the third iteration:
    3.1. Raster scans each labelled background region in the text image,
    3.2. If all the background region is surrounded with foreground pixels,
    3.3. Relabel the surrounding foreground pixels with the label "6".
```

C. Candidate Segmentation Points Detection

I. Modified vertical projection histogram

Vertical projection histogram (VPH) is a popular approach for the detection of the segmentation points. It counts the foreground pixels in each column as follows:

$$VerticalProjectionHistogram\ (y) = \sum_x P(x, y). \qquad (2)$$

Areas with lower pixel density are identified as candidate segmentation points (CSPs), as shown in Figure16 (a). One disadvantage of using the original text image is the excessive number of CSPs. To overcome this disadvantage, an attempt was made to segment the Arabic script proposed by Al Hamad and Abu Zitar (2010) by using VPH based on the text skeleton, instead of the original image, thereby minimizing the number of CSPs (Figure 16(b)). However, this technique suffers from three main weaknesses. First, the algorithm is not capable of distinguishing between the "ligatures" and other strokes. Second, the algorithm is not capable of recognizing some of the characters shapes that appear as ligatures in the histogram. Third, the algorithm is not capable of detecting characters and words/sub-words that overlap. Therefore, a modification of VPH is proposed to overcome all the weaknesses by using the labelled text skeleton after applying the SLDF2 instead of the original text image as shown in Figure 16(c).

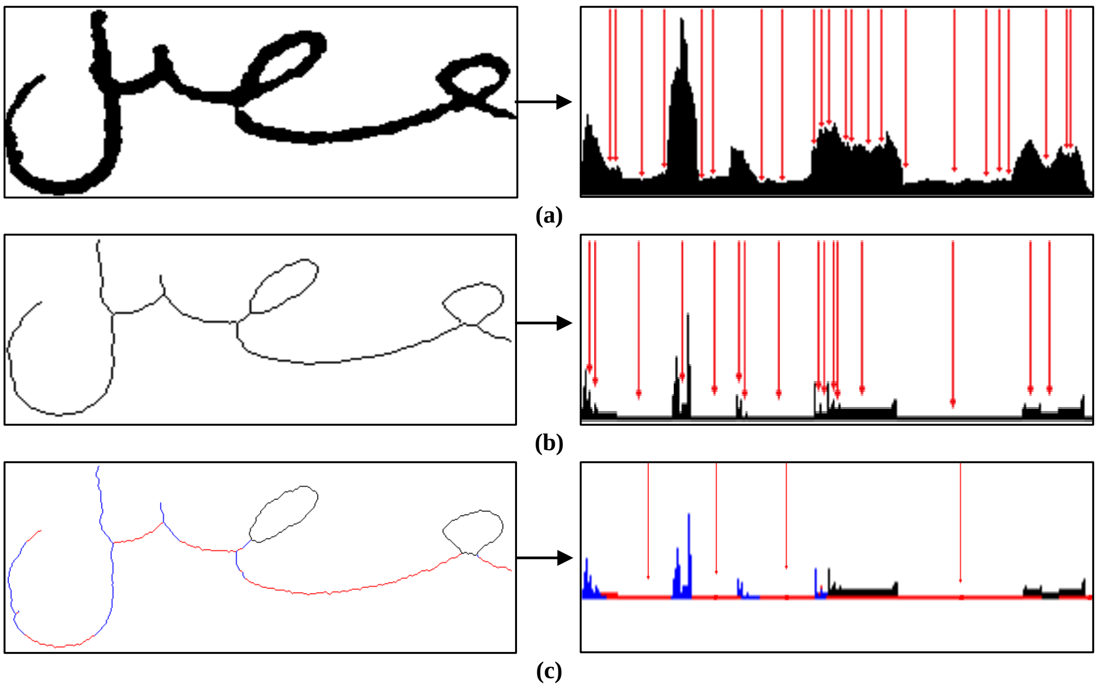


FIGURE 16. (a) Vertical projection histogram (VPH) of original text, (b) Vertical projection histogram of text skeleton using Al Hamad and Zitar's technique and (c) The proposed modified vertical projection histogram of the labeled text skeleton

II. Words/sub-words overlapping

Text words are implemented primarily from a set of connected foreground pixels. Each group represents either a word, sub-word, supportive object or even noise. Using the VPH in the text segmentation process leads to the false detection of segmentation points in cases where words/sub-words are vertically overlapped. This problem can be overcome using a connected-component-labelling algorithm to detect these components and calculate the proposed MVPH for each of them separately.

III. Missed character (fake ligature) detection

Ligatures are the horizontal primitives that are used to connect the characters in Arabic script. In some people's handwriting styles, the part of the text that is supposed to represent a vertical stroke appears instead as a horizontal stroke (a ligature) in the representation of the labeled text skeleton. These forgery ligatures take a hump (bell) shape, as shown in Figure 16. In this research, an algorithm for detection such fake ligatures are proposed as follows:

Scan each component of the text labelled skeleton from the left-top corner, row-by-row

1. Detect the concatenated pixels labeled "4" that do not share the vertical coordinate with any other foreground pixels in the same component.
2. Check if there are at least two height pixels (labeled "4") that form a peak.

If such a case exists, label all the pixels that form the peak with the value "2".

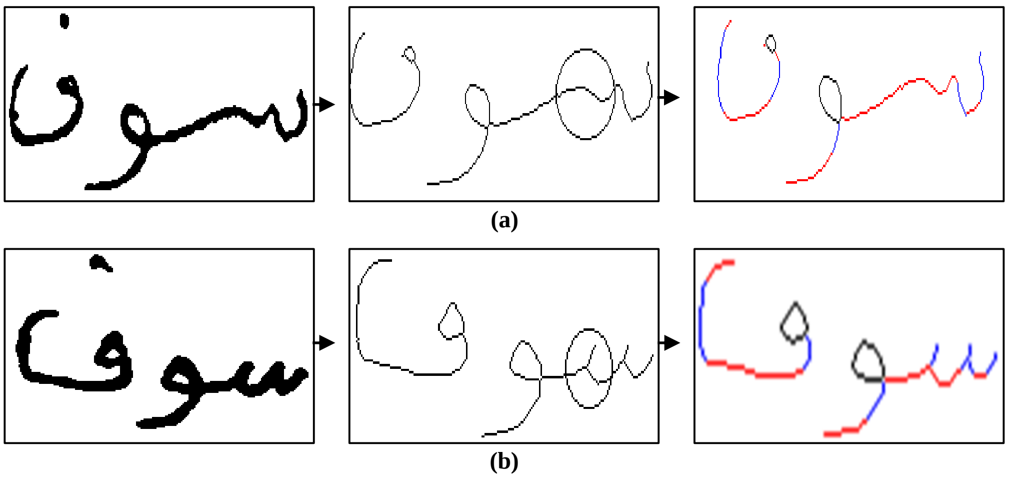


FIGURE 17. Example of word سوف with different writing styles (a) missing a vertical stroke, (b) not missing a vertical stroke

An example of the missing character detection is shown in Figure 17.

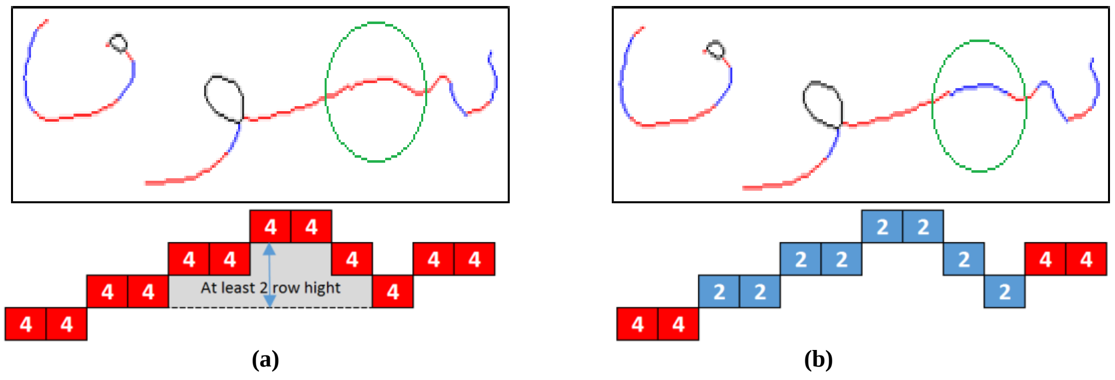


FIGURE 18. Example of the "missing vertical stroke detecting process" (a) before (b) after

IV. Overlapped characters' detection

In Arabic script, overlapped characters exist if one of four characters, ي ،هـ ،م ،ح, are connected vertically in the same connected component, as shown in Figure 18.

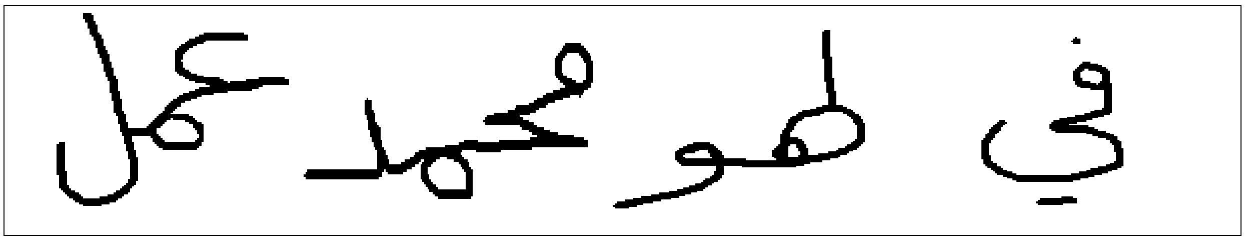


FIGURE 19. Examples of characters overlapping in Arabic script

In this research, the proposed method is able to detect the overlapped characters that consist of at least one loop (ح، ـمـ، ـهـ). The following is the procedure:

Case A

1. In the labelled skeleton, look for the existence of pixels labelled “6” (loop),
2. Check if the upper side of the loop is connected with any pixel labelled “2” [vertical connection point VCP],
3. Check if the left side of the loop is connected with any pixel labelled “4” [horizontal connection point HCP],
4. Compute D_1& D_2 as follows:

- D1= the vertical distance between the *x*-coordinate of the higher pixel of the loop and the HCP,
- D2= the vertical distance between the *x*-coordinate of the lower pixel of the loop and the HCP,

5. If D2 $\geq$ D1, then mark the VCP as an actual segmentation point (ASP).

An example of the first case of overlapped characters’ detection is shown in Figure 20 (a, b).

Case B

1. In the labelled skeleton, look for the existence of pixels labelled “6” (loop),
2. Check if any part of the loop shares the vertical coordinate with at least two layers of pixels labelled “4” under the loop,
3. Mark the middle pixel of the higher layer labelled “4” under the loop as an ASP.

An example of the second case of overlapped characters’ detection is shown in Figure 20 (c).

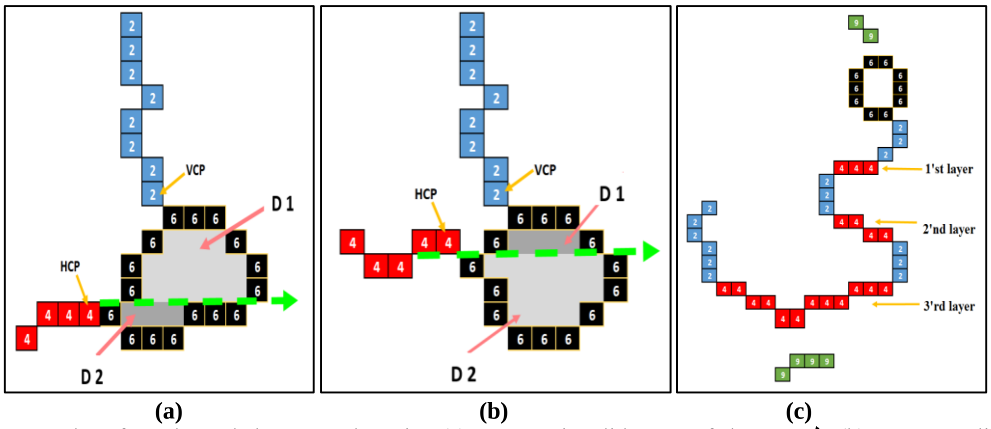


FIGURE 20. Examples of overlapped characters detection (a) case A: “invalid CSP” of character ط, (b) case A:“valid CSP” of subword لمـ, and (c) case B: “valid CSP” of word في

CHARACTERS SEGMENTATION PROCESS

In this research, the vertical projection histogram is modified by applying it onto the gained labelled skeleton from the proposed SLDF2 technique instead of the original image (Figure 21 (a). The following are the steps of the candidate segmentation point (CSP) detection process. In the modified vertical projection histogram (MVPH), look for the existence of the adjacent columns that contain only one-pixel height labelled “4”, as shown in Figure 18(b). In the labelled skeleton, detect the corresponding regions that present the adjacent columns from the prior step. Each region is considered a ligature, and the CSP is located in the middle of the corresponding original text.

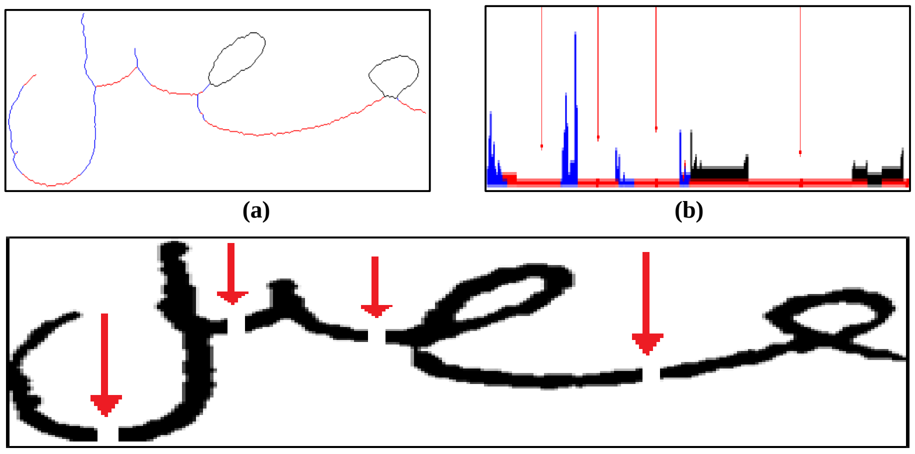

(a) (b) (c)

FIGURE 21. Example of the candidate segmentation point (CSP) detection process: (a) text representation after applying the proposed SLDF2, (b) the modified VPH of the text representation, and (c) the CSP detection result.

SEGMENTATION POINT VALIDATION

A set of Arabic script structural\topological rules are proposed for validating the candidate segmentation points. These rules are presented in the form of cases that include most popular cases.

Case A: *The component starts with a horizontal stroke*

In this case, many characters start with a horizontal stroke (the right-most part). Some of these strokes are not part of a character nor a connection stroke between two characters (Figure 22). The procedure of distinguishing between real and fake connection strokes is as follows:

1. Scan the modified VPH of each component (word/sub-word),
2. If one-pixel-height adjacent columns where the pixels are labeled “4” (horizontal stroke) that contain a CSP exist in the right-most part of the histogram (the horizontal stroke is not connected with the other strokes from the right side): go to step 3,
3. Scan the corresponding labeled skeleton,
4. If the horizontal stroke shares the vertical coordinate interval with any supportive object then state the CSP as valid.

where the vertical coordinate interval = [$Y_{\_Coordinate}$ of the examined CSP ± ½ (Distance between the examined CSP and the first left CSP)], if the number of CSPs in the examined component >1. The vertical coordinate interval = [$Y_{\_Coordinate}$ of the most left of component to the $Y_{\_Coordinate}$ of the most right of component], if the number of CSPs in the examined component =1.

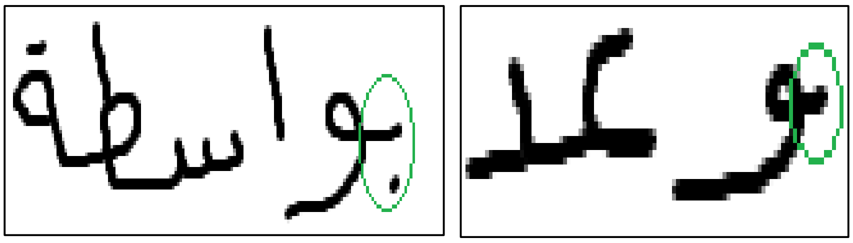

(a) (b)

FIGURE 22 . Examples of (a) valid CSP, and (b) invalid CSP in the case where the component starts with a horizontal stroke

Case B: *The component ends with a horizontal stroke*
Many characters in Arabic script, such as (و، هـ، ظ، ز، ر، ذ، د), end with a horizontal stroke (the left-most part), which contains a CSP. In fact, they are all invalid segmentation points. The procedure of detecting is as the following:

1. Scan the modified VPH of each component (word/sub-word),
2. If one-pixel-height adjacent columns where the pixels are labeled as “4” (horizontal stroke) that contain a CSP exist in the left-most part of the histogram (the horizontal stroke is not connected with the other strokes from the left side): then state, the CSP as invalid.

Case C: *The component ends with a horizontal stroke that is concatenated with a vertical stroke from the left and a hole from the right*
In this case, the left-most part of the component is in the order of: the horizontal stroke (containing a CSP) connected with a vertical stroke from the left side and with a loop from the right side. The vertical stroke is not connected with any other stroke from the left, such as (طا، فا، ما، م). The examined CSPs are not always valid, such as (م). The steps of distinguishing the valid CSP are as follows:

1. In the labelled skeleton, look for the existence of a horizontal stroke that contains a CSP and connects with a loop from the right side and a vertical stroke from the left side.
2. Compute the following:
    a. D_1= (higher-pixel $_{\text{Y\_coordinateof the vertical stroke}}$)-(CSP$_{\text{Y\_coordinate}}$).
    b. D_2 = (CSP$_{\text{Y\_coordinate}}$)-(lower-pixel $_{\text{Y\_coordinateof the vertical stroke}}$).

3. If $D_1>(2 \times D_2)$, then state the CSP as valid.

An example of this case is shown in Figure 23.

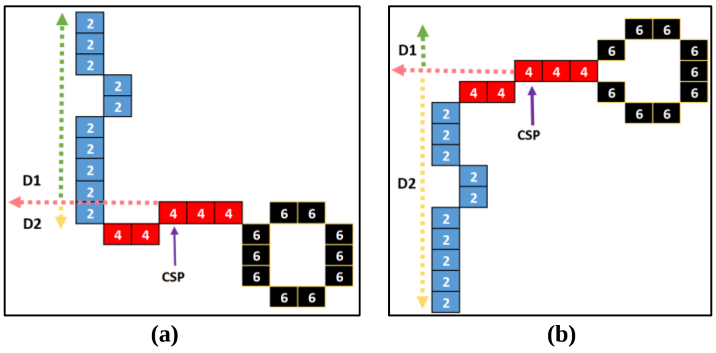


FIGURE 23. Example of: (a) valid CSP, and (b) invalid CSP

Case D: *The component ends with a horizontal stroke that is concatenated with two vertical strokes, one from the left and the other from the right.*
In this case, the left-most part of the component is in the order: horizontal stroke (containing a CSP) connected with two vertical strokes (from the left side and from the right side). The vertical stroke in the left is not connected with any other stroke, such as (ن، ت، يا). The examined CSP are not always valid, such as (ل). The steps of distinguish the valid and invalid CSP are as follows:

1. Scan the labelled skeleton, check if the component ends with a horizontal stroke that is concatenated with two vertical strokes (one from the left and the other from the right).
2. Compute the following:
    a. D_3 = (higher-pixel$_{\text{Y\_coordinate of the right vertical stroke}}$) -(CSP$_{\text{Y\_coordinate}}$).
    b. D_4 = (higher-pixel$_{\text{Y\_coordinate of the left vertical stroke}}$) - (CSP$_{\text{Y\_coordinate}}$).
3. If $D_4 > (2 \times D_3)$, then state the CSP as valid; else, state as invalid.

An example of this case is shown in Figure 24.

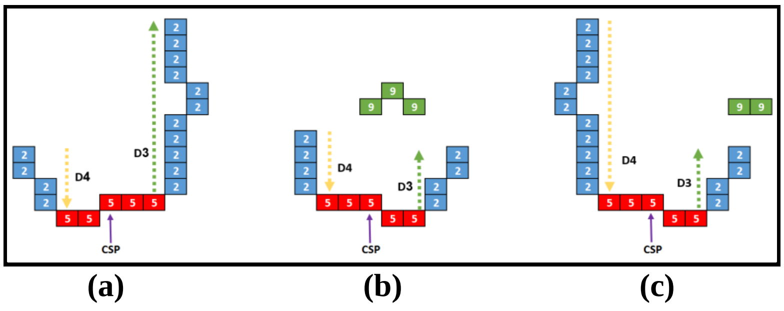


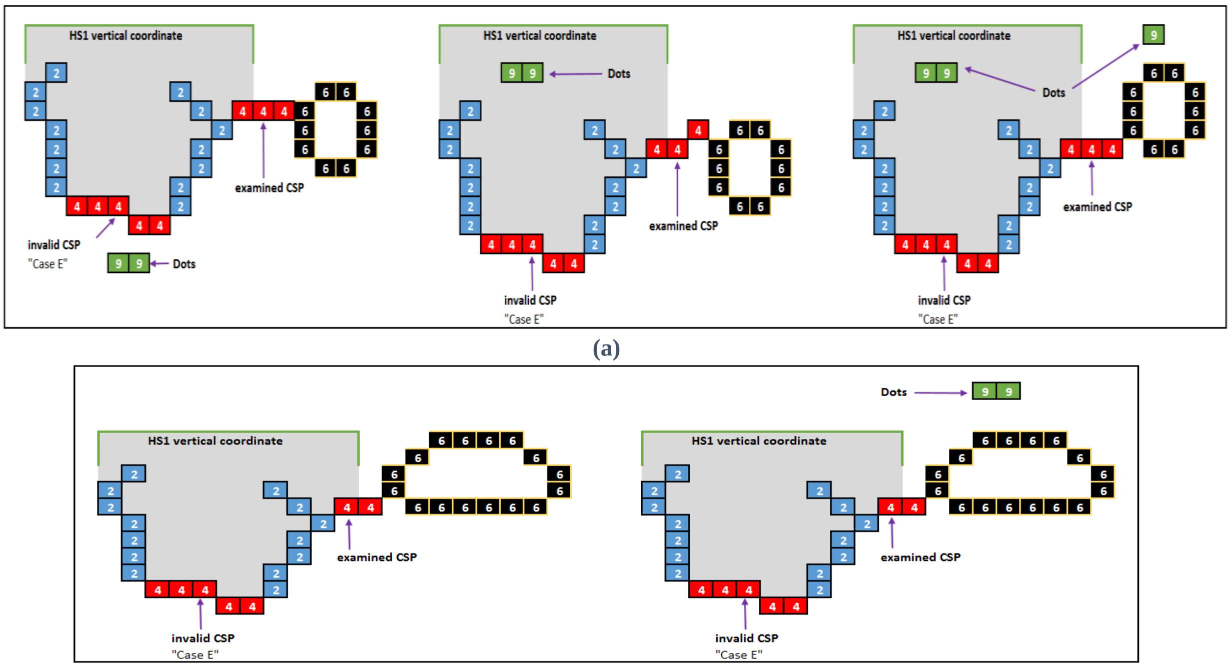
FIGURE 24. Example of: (a and b) invalid CSP, and (c) valid CSP that matches case D

Case E: *The component ends with a set of concatenated labelled strokes in order: vertical stroke, horizontal stroke, vertical stroke, horizontal stroke and loop from left to right, consequently.*
In this case, all the CC's end with, in order (from left to right): first vertical stroke (VS1), first horizontal stroke (HS1), second vertical stroke (VS2), second horizontal stroke (HS2) and Loop, such as (ص، ض، من، مي، ف). HS2 has a CSP that is examined as follows:

Scan the labelled skeleton, check if HS1 shares the vertical interval with dots, and then the examined CSP is valid (else it is invalid), where the HS1 vertical interval = from (most left pixel of VS1$_{\text{Y\_ coordinate}}$), to (CSP$_{\text{Y\_ coordinate}}$). An example of this case is shown in Figure 22.

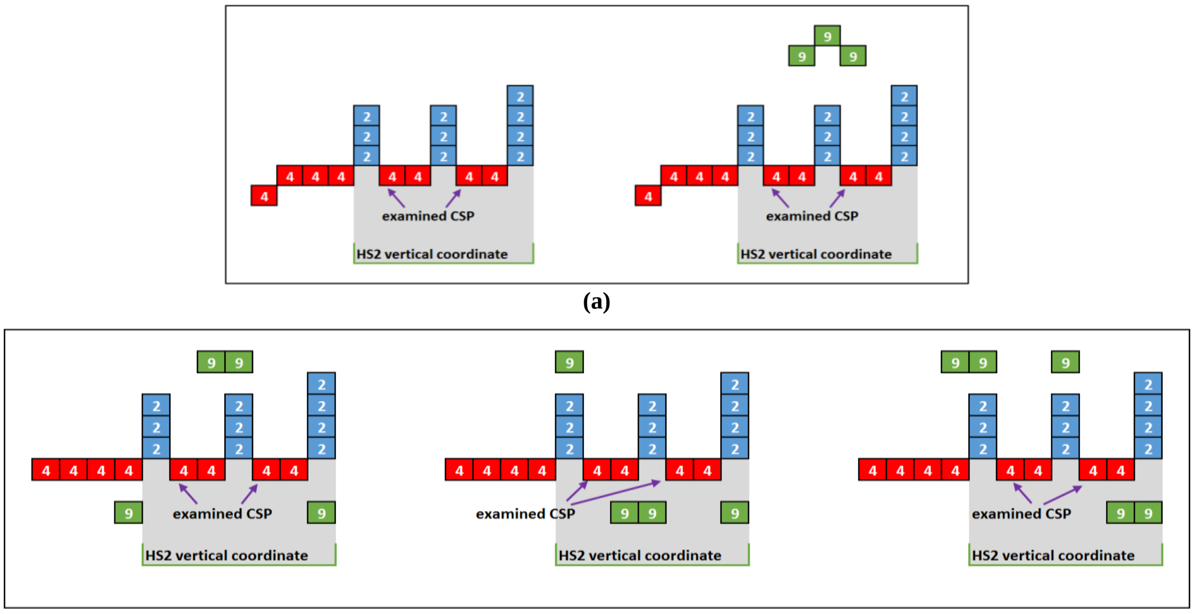
Case F: *The component contains a set of concatenated labelled strokes, in order: three vertical strokes (VS1, VS2, and VS3) connected by two horizontal strokes (HS1, and HS2).*
In this case, the examination procedure is as follows:

In the labelled skeleton, check if there is any dot cited in the lower HS2 vertical interval; if yes, both CSPs are valid; otherwise, they are invalid, where the lower HS2 vertical coordinate = from (most left pixel of VS1$_{Y\_ \text{coordinate}}$) to (rightest pixel of VS3$_{Y\_ \text{coordinate}}$) under the HS2. An example of case F is shown in Figure 25 and Figure 26.



(b)

FIGURE 25. Examples of Case E: (a) valid CSP [فن,من, مي] (b) invalid CSP [ضـ ، صـ]



(b)

FIGURE 26. Examples of Case F:(a) invalid CSP [شـ، سـ] (b) valid CSP [بيـ، بنبـينت،]

EXPERIMENTAL RESULTS AND DISCUSSION

This section outlines the results obtained by using the proposed stroke labelling based on direction features (SLDF1 and SLDF2) techniques and the character segmentation method. The Arabic Center for Document Analysis and Recognition (ACDAR) handwritten dataset (Al Hamad & Abu Zitar 2010; Al Hamad & Hamdi-Cherif 2012) is designed for the Arabic document analysis domain. It contains 500 random words that where extracted from two paragraphs covering all shapes of Arabic characters, written by 10 writers (12-50 years in age). The total number of characters in the database is 2562. In this research, the ACDAR dataset was used for a majority of character segmentation experiments. Figure 27 shows samples of the ACDAR dataset.

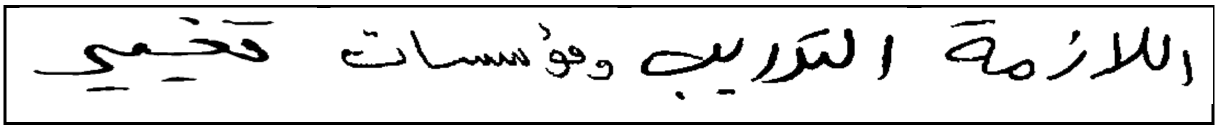


FIGURE 27. Samples from ACDAR dataset (Husam, Al Hamad & Hamdi-Cherif 2012) used in experiments

STROKE LABELLING BASED ON DIRECTION FEATURES (SLDF) TECHNIQUE EVALUATION

The results from the proposed SLDF1 technique were comparatively evaluated with results from both techniques by Al Hamad and Abu Zitar (2010) and Blumenstein et al. (2007). The number of processed pixels needed to accomplish the strokes labelling for all 500 word images of the ACDAR data set were used to determine technique speed using the following equations:

$$\begin{aligned} NPL &= NFP \times NIRI, \\ NPN &= NSP \times NIRI, \\ TNPP &= NPL + NPN. \end{aligned} \tag{3}$$

where *NPL* is the Number of Pixels involved in the determining value (Labelling) stage; *NFP* is the Number of Foreground Pixels; *NIRI* is the Number of Iterations of Reading (scanning) the image; *NPN* is the Number of Pixels involved in the Normalization stage; *NSP* is the Number of Spurious Pixels; and *TNPP* is the Total Number of Processed Pixels in both the determining values and normalization stages.

Blumenstein et al.'s technique is fast in assigning the direction labels (the number of foreground pixels involved in the determining values stage) because it scans the entire image only one time (one iteration), where the current pixel label depends on the direction of the previous labelled pixel. However, the labelling stage result suffers from an extreme number of spurious pixels that need normalization. The normalization stage takes two iterations to read the image pixels and perform the normalization.

The technique established by Al Hamad and Abu Zitar (2010) modifies the process of determining the direction labels of pixels to minimize the spurious pixels by scanning all of the images about four times whereby one time for each direction (vertical, horizontal, right diagonal, and left diagonal). However, the overall processing period is high due to repetitive times of image scanning process in both the labelling and normalization stages. The proposed technique scans the image one time to assign the label of each pixel and another time to normalize the spurious pixels by detecting their relationship of 8 neighbouring pixels and changing their labels in the same iteration subsequently.

EVALUATION METHOD

This section discusses the effectiveness of the proposed character segmentation method called SLDF2. Due to unavailability of a benchmark dataset with a ground truth table that provides the location of the correct segmentation points, an expert validated the segmentation results. A quantitative analysis was conducted for the proposed method's results. The following equations of the Precision, Recall, Accuracy and F-measurements were used to determine segmentation accuracy:

$$Recall = \frac{TP}{\mathrm{TP} + \mathrm{FN}} \,,$$
$$Precision = \frac{TP}{\mathrm{TP} + \mathrm{FP}} \,,$$
$$Accuracy = \frac{TP}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN}} \,,$$
$$F - Measure = \frac{2 \times Recall \times Precision}{(Recall + Precision)} \,. \tag{4}$$

The dataset is divided into 10 sets (writers). For each set, we specify the true positive value (*TP*) as the correct detected segmentation point, the false negative (*FN*) as undetected all segmentation points (missed/under-segmentation), and the false positive (*FP*) as more than one undetected segmentation point for each character (over-segmentation). Then, we calculate the overall Recall (*RC*), Precision (*PR*), and F-measures (*FM*) of the proposed method as in Equation 2. Table 2 below shows the results for all competing labelling techniques.

TABLE 2. Detailed results of (a) Blumenstein et al.'s labelling technique, (b) Al Hamad and Zitar's labelling technique, (c) the SLDF1 labelling technique, and (d) an average results of all competing labelling techniques

	(a)			(b)			(c)		
Writer	**NPL**	**NPN**	**TNPP**	**NPL**	**NPN**	**TNPP**	**NPL**	**NPN**	**TNPP**
1	22,604	12,002	34,606	90,416	1,884	92,300	22,604	942	23,556
2	25,354	12,644	37,998	101,416	2,190	103,606	25,354	1,095	26,456
3	30,072	15,450	45,522	120,288	2,398	122,686	30,072	1,199	31,295
4	19,141	10,168	29,309	76,564	1,302	77,866	19,141	651	19,799
5	24,108	12,580	36,688	96,432	2,020	98,452	24,108	1,010	25,146
6	24,612	13,462	38,074	98,448	2,340	100,788	24,612	1,170	25,801
7	22,119	12,708	34,827	88,476	1,816	90,292	22,119	908	23,034
8	29,170	13,688	42,858	116,680	2,174	118,854	29,170	1,087	30,267
9	32,577	16,762	49,339	130,308	2,304	132,612	32,577	1,152	33,737
10	25,390	11,310	36,700	101,560	1,830	103,390	25,390	915	26,298
Total	255,147	130,774	385,921	1,020,588	20,258	1,040,846	255,147	10,129	265,389

Method	**NPL**	**NPN**	**TNPP**	**Number of iteration (image scanning)**
Blumenstein	255,147	130,774	385,921	3 (1 for labelling + 2 for normalization)
Al Hamad	1,020,588	20,258	1,040,846	6 (6 for labelling + 2 for normalization)
Proposed	255,147	10,129	265,389	2 (1 for labelling + 1 for normalization)

(d)

TABLE 3. Results summary of the proposed character segmentation method

Writer	Actual segmentation points	TP	FN	FP	PR	RC	Accuracy	FM
1	348	331	13	4	96.11%	98.81%	95.01%	97.44%
2	332	300	16	16	94.96%	94.96%	90.40%	94.96%
3	387	338	28	21	92.44%	94.15%	87.42%	93.29%
4	309	271	33	5	89.02%	98.27%	87.65%	93.42%
5	302	275	20	7	93.07%	97.58%	90.97%	95.27%
6	353	329	16	8	95.29%	97.59%	93.10%	96.43%
7	312	303	8	1	97.35%	99.61%	96.98%	98.47%
8	350	321	13	16	96.13%	95.39%	91.86%	95.76%
9	355	344	6	5	98.42%	98.57%	97.03%	98.49%
10	301	283	17	1	94.35%	100.00%	94.35%	97.09%
Total	3349	3096	170	83	94.71%	97.49%	92.48%	96.06%

The following measurements were calculated using the proposed segmentation method: PR = 94.71%, RC = 97.49%, Accuracy= 92.48% and *F*-measure = 96.06%, as reported in Table 3. Sample images from the dataset and segmentation points detected using the proposed method are displayed in Figure 28.

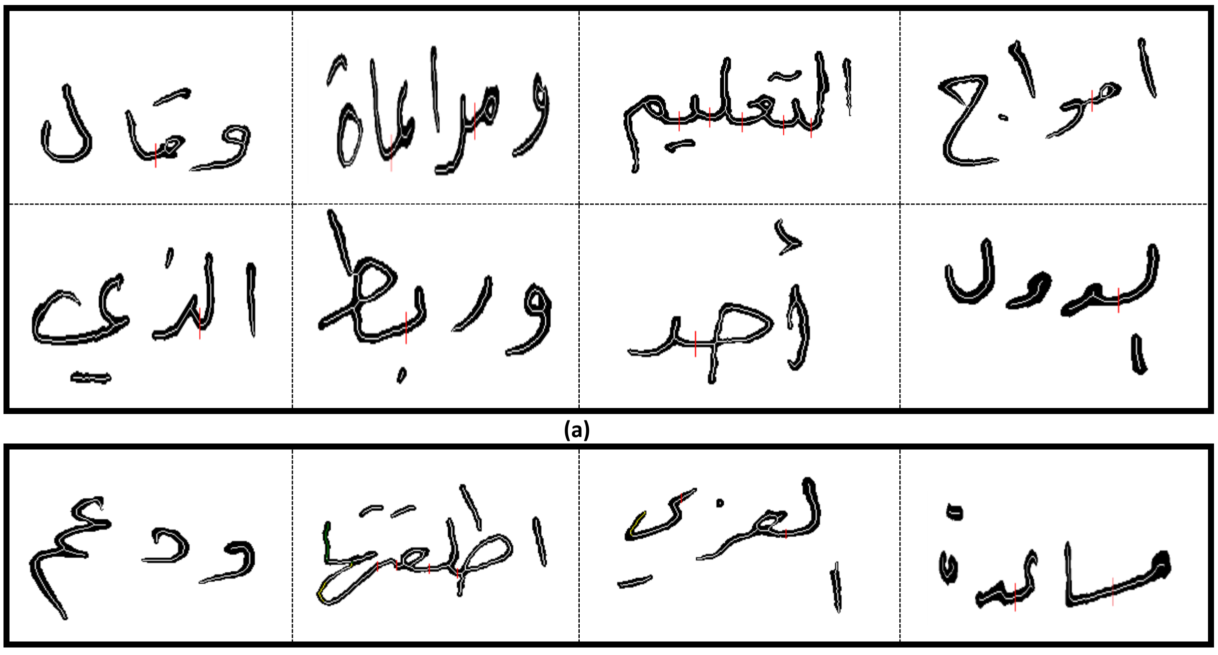


FIGURE 28. Segmentation points (SP) detected by the proposed method (a) Correct SP, and (b) missed SP

COMPARISON BETWEEN THE PROPOSED AND AL HAMAD AND ZITAR'S CHARACTER SEGMENTATION METHODS RESULTS

The average correct segmentation accuracy of the proposed segmentation method was 92.48%. We compared our proposed method with Al Hamad and Zitar's method presented in Al Hamad and Abu Zitar (2010), where they used the same dataset. Their proposed segmentation method is neural-based to validate the segmentation points detected via vertical projection histogram of the text skeleton with 82.98% average correct segmentation accuracy.

Table 4 and 5 show a results summary of the Al Hamad and Zitar's method and the comparison results between the proposed ACS-SLDF2 and the Al Hamad and Zitar's methods. The findings justify stating that the proposed method outperforms the Al Hamad and Zitar's method. The proposed method is able to detect segmentation points based on SLDF2 and the modified vertical projection histogram, which leads to producing fewer candidate segmentation points, which are validated via a set of proposed language structural-rules. Different from the proposed method, the Al Hamad and Zitar's method detects segmentation points based on the text skeleton and modified vertical projection histogram, which leads to producing more candidate segmentation points, which are validated via a set of direction features and neural networks.

TABLE 4. Results summary of the Al Hamad and Zitar's character segmentation method

Writer	Actual segmentation points	TP	FN	FP	PR	RC	Accuracy	FM
1	348	297	18	34	89.73%	94.29%	85.10%	91.95%
2	332	269	14	49	84.59%	95.05%	81.02%	89.51%
3	387	318	11	60	84.13%	96.66%	81.75%	89.96%
4	309	270	10	29	90.30%	96.42%	87.37%	93.26%
5	302	247	23	32	88.53%	91.48%	81.79%	89.98%
6	353	284	20	49	85.29%	93.42%	80.45%	89.17%
7	312	263	10	40	86.80%	96.34%	84.03%	91.32%
8	350	284	28	38	88.20%	91.03%	81.14%	89.59%
9	355	284	12	59	82.80%	95.95%	80.00%	88.89%
10	301	263	14	25	91.32%	94.95%	87.09%	93.10%
Total	3349	2779	160	415	87.17%	94.56%	82.98%	90.67%

TABLE 5. Comparison of results between the proposed and Al Hamad and Zitar's character segmentation methods

	Correct Segmentation		Under-segmentation		Over-segmentation	
Writer	**Proposed**	**Al Hamad-Zitar**	**Proposed**	**Al Hamad-Zitar**	**Proposed**	**Al Hamad-Zitar**
1	94.98%	85.34%	3.85%	4.89%	1.43%	9.77%
2	90.40%	81.02%	4.80%	4.22%	4.80%	14.76%
3	87.30%	82.17%	7.14%	2.33%	5.56%	15.5%
4	87.64%	87.38%	10.81%	3.24%	1.54%	9.39%
5	90.98%	81.79%	6.77%	7.28%	2.26%	10.39%
6	93.10%	80.45%	4.60%	5.67%	2.30%	13.88%
7	96.98%	84.29%	2.64%	2.88%	0.38%	12.82%
8	91.85%	81.14%	3.70%	8%	4.44%	10.86%
9	96.89%	80%	1.56%	3.38%	1.56%	16.62%
10	94.35%	87.38%	5.65%	4.32%	0.01%	8.31%
Average	92.48%	82.98%	5.15%	4.6%	2.43%	12.42%

STROKE LABELLING BASED ON DIRECTION FEATURES RESULTS DISCUSSION

In this section, the results of stroke labelling based on direction features techniques were compared under many measurement criteria:

1. Number of foreground pixels involved in determining the direction value (labelling) stage,

2. Number of iterations needed to complete the labelling stage,
3. Number of foreground pixels involved in normalization of the spurious pixels stage, and
4. Number of iterations needed to complete the normalization stage.

Based on the measurement criteria results of the techniques of Blumenstein et al. (2007), Al Hamad and Abu Zitar (2010), and our proposed SLDF1, it is clear that the proposed SLDF1 technique is superior in performance as follows:

1. Blumenstein et al.'s technique labels the strokes of the text image in one iteration, which make the number of pixels involved in the labelling stage equal to the number of actual text image foreground pixels. However, the technique suffers from an extreme number of spurious pixels due to the used labelling technique where each foreground pixel takes the label value based on the direction of its eight neighbouring former labelled pixels. Moreover, the technique takes two iterations to normalize the spurious pixels (one for detecting the spurious pixels and one for changing the pixel label) which make the pixels involved in this stage equal to double the number of spurious pixels.
2. Al Hamad and Zitar's technique labels the strokes of the text image in four iterations to minimize the number of spurious pixels, where each foreground pixel takes a value based on the direction of its 8 neighbouring foreground pixels in a predefined direction priority (one iteration for each direction). However, the technique suffers from an extreme number of pixels involved in the labelling stage, which is equal to four times the number of text image foreground pixels. Moreover, the technique takes two iterations to normalize the spurious pixels (one for detecting the spurious pixels and one for changing the pixel labels) which makes the pixels involved in this stage equal to double the number of spurious pixels.
3. Our proposed SLDF1 technique was able to label the strokes of the text image in one iteration, which makes the number of pixels involved in the labelling stage equal to the actual number of text image foreground pixels, where each foreground pixel takes a value based on the direction of its eight neighbouring foreground pixels in a predefined priority in one single iteration. Moreover, the technique takes only one iteration to normalize the spurious pixels by detecting the spurious pixels and changing the pixel label based on the same predefined direction priority used in the labelling stage.

ARABIC CHARACTER SEGMENTATION RESULTS DISCUSSION

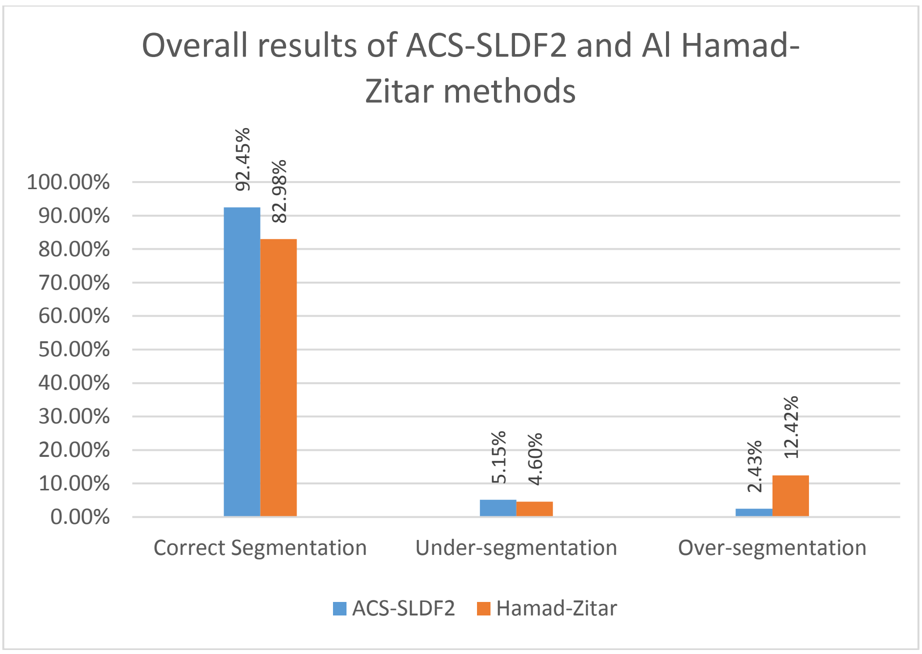
Comparing the handwritten character segmentation results with that of other researchers in the literature is not a simple task. The central complications that arise are the variances in experimental approach, experimental settings, and the experimental dataset. Generally, most Arabic character segmentation methods are part of character recognition systems, where the system accuracy is measured in terms of the entire character recognition system instead of just the character segmentation method.

In this section, the results of Arabic character segmentation methods of Husam, Al Hamad and Abu Zitar (2010) and the proposed ACS-SLDF2 were compared under many measurement criteria, correct segmentation accuracy rate, under-segmentation rate, over-segmentation rate, words/sub-words and characters overlapping avoidance capability and missed character detection success rate. To present a clear evaluation of the proposed ACS-SLDF2 and Al Hamad-Zitar methods, both of them are applied on the ACDAR dataset (Al Hamad & Hamdi-Cherif 2012). As shown in Figure 29, the proposed ACS-SLDF2 method outperforms the Al Hamad-Zitar method based on the measurements criteria results as follows:

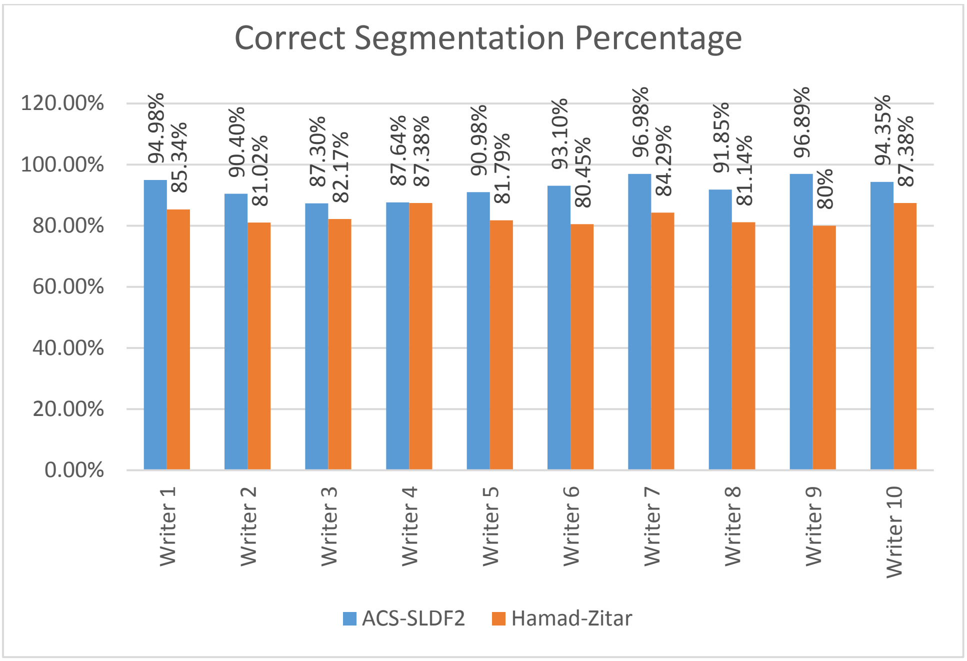
1. The Al Hamad and Zitar's method was able to detect the correct segmentation points of the ACDAR dataset with a 82.98% success rate. The achieved accuracy rate is slightly low due to multiple reasons: First, the method suffers from the words/sub-words

overlapping and characters overlapping problems. Moreover, it suffers from many popular cases of missed characters (that are caused by using the text skeleton to present the vertical projection histogram) where some characters are represented by a horizontal stroke instead of a vertical stroke. All of these disadvantages increase the undesired percentage rates of over-segmentation and reduce the overall correct segmentation percentage rate.

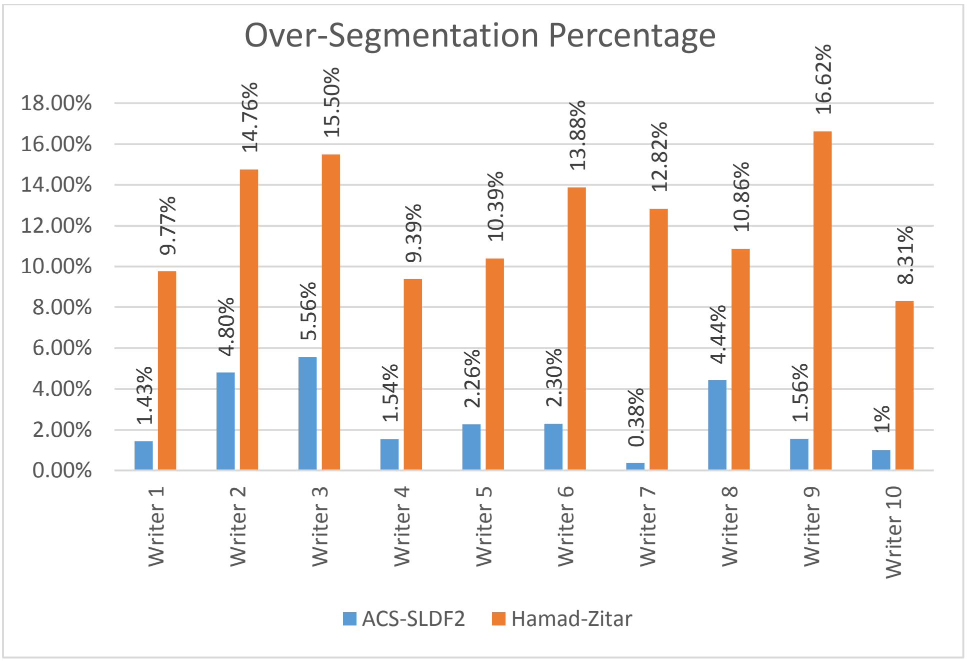
2. Our proposed ACS-SLDF2 method was able to detect the correct segmentation points of the ACDAR dataset with a 92.48% success rate. The high accuracy rate is achieved due to multiple reasons: First, the proposed method overcomes the words/sub-words overlapping problem by separating each of them using a connected component labelling technique before starting the segmentation process. Moreover, it can detect many popular cases of characters overlapping and detect the missed characters (that are caused by using the text skeleton to present the vertical projection histogram) by taking advantage of using the text skeleton after applying the direction features, which reduce the under-segmentation percentage rate. In addition, many character segmentation point validation rules are proposed to eliminate invalid segmentation points. All of these advantages reduce the undesired percentage rates of both under- and over-segmentation.



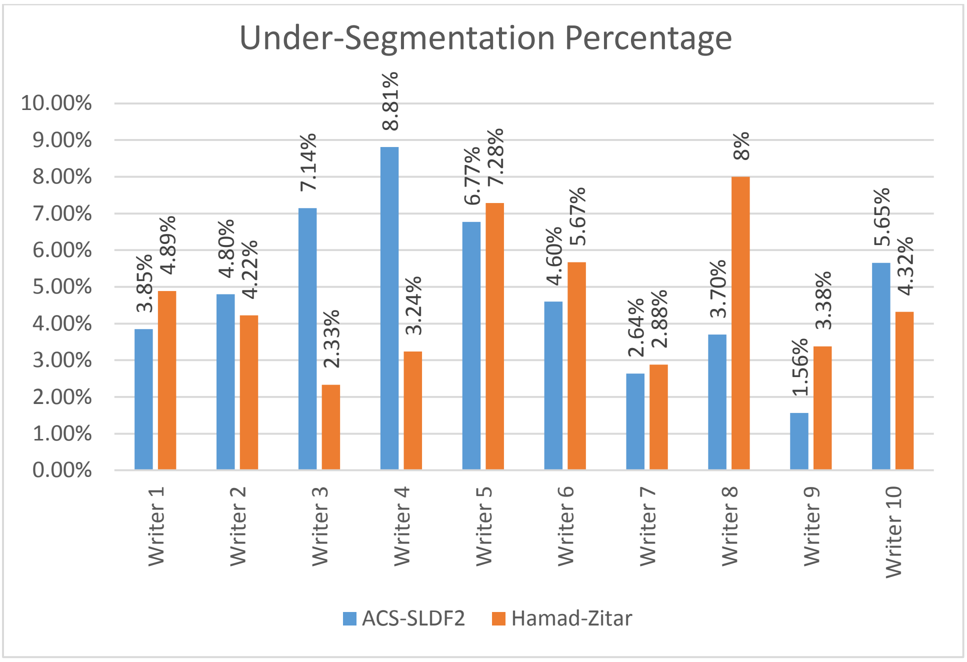
(a)


Correct Segmentation Percentage
120.00%
100.00%
80.00%
60.00%
40.00%
20.00%
0.00%
94.98%
85.34%
90.40%
81.02%
87.30%
82.17%
87.64%
87.38%
90.98%
81.79%
93.10%
80.45%
96.98%
84.29%
91.85%
81.14%
96.89%
80%
94.35%
87.38%
Writer 1
Writer 2
Writer 3
Writer 4
Writer 5
Writer 6
Writer 7
Writer 8
Writer 9
Writer 10
ACS-SLDF2
Hamad-Zitar

(b)


Over-Segmentation Percentage
18.00%
16.00%
14.00%
12.00%
10.00%
8.00%
6.00%
4.00%
2.00%
0.00%
1.43%
9.77%
4.80%
14.76%
5.56%
15.50%
1.54%
9.39%
2.26%
10.39%
2.30%
13.88%
0.38%
12.82%
4.44%
10.86%
1.56%
16.62%
1%
8.31%
Writer 1
Writer 2
Writer 3
Writer 4
Writer 5
Writer 6
Writer 7
Writer 8
Writer 9
Writer 10
ACS-SLDF2
Hamad-Zitar

(c)



(d)

FIGURE 29. Comparison between the proposed ACS-SLDF2 and Al Hamad-Zitar methods (a) summary of the overall results, (b) detailed correct segmentation percentage rates for each writer, (c) detailed over-segmentation percentage rates for each writer, and (d) detailed under-segmentation percentage rates for each writer

CONCLUSION

An Arabic handwritten character segmentation method is introduced in this paper. It consists of many phases, such as labelling the text skeleton, detecting the candidate segmentation points (CSPs) via a modified vertical projection histogram, and verifying the validity of CSPs based on proposed language structural-rules. In addition, the SLDF1 algorithm offers various improvements, including decreased execution time and a reduction of the labelling values to only two values. An experiment was conducted to show the performance of our method in solving several challenges in Arabic text segmentation, such as characters and sub-words overlapping. Experimental results of the proposed stroke labelling technique and segmentation method are promising. The following segmentation accuracies were achieved using the proposed method: PR = 97.45%, RC = 94.73% and *F*-measure = 96.07%. Excellent writing is an impact factor in succeeding in the character segmentation process and in reducing the error rate, as well. In addition, avoiding overlapping characters in writing plays a significant role in increasing the overall correct segmentation rate.

ACKNOWLEDGMENTS

This research is funded by the Universiti Kebangsaan Malaysia entitled AP-2017-005/2 and FRGS-1-2019-ICT02-UKM-02-9. The researchers would also like to thank the Universiti Kebangsaan Malaysia Lab for their support of this research.

REFERENCES

Abdulla, S., Al-Nassiri, A., & Salam, R. A. 2008. Off-Line Arabic Handwritten Word Segmentation Using Rotational Invariant Segments Features. *International. Arab Journal of Information. Technology*, 5(2): 200-208.

Abdullah, M. A., Al-Harigy, L. M., & Al-Fraidi, H. H. 2012. Off-line arabic handwriting character recognition using word segmentation. *Journal of Computing*, 4(3): 40-44

Abu-Ain, T., Abdullah, S. N. H. S., Bataineh, B., & Omar, K. 2013. A fast and efficient thinning algorithm for binary images. *Journal of ICT Research and Applications*, 7(3): 205-216.

Ahmad, R., Naz, S., Afzal, M. Z., Rashid, S. F., Liwicki M. and Dengel, A. 2017. KHATT: A Deep Learning Benchmark on Arabic Script," *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, Kyoto, 2017, pp. 10-14.

Ahmed, S.B., Naz, S., Razzak, M.I., and Yusof, R.B. 2019. A Novel Dataset for English-Arabic Scene Text Recognition (EASTR)-42K and Its Evaluation Using Invariant Feature Extraction on Detected Extremal Regions. *IEEE Access*, 7(8641268): 19801-19820.

Ali, A.A.A., Suresha, M. 2019. Efficient Algorithms for Text Lines and Words Segmentation for Recognition of Arabic Handwritten Script. *Advances in Intelligent Systems and Computing*, 882: 387-401.

Al-Badr, B., & Haralick, R. M. 1995. Segmentation-free word recognition with application to Arabic. *Proceedings of 3rd International Conference on Document Analysis and Recognition, Montreal, Quebec, Canada*, vol.1, pp. 355-359

Al-Helali, B. M., & Mahmoud, S. A. 2017. Arabic Online Handwriting Recognition (AOHR): A Survey. *Journal ACM Computing Surveys (CSUR)*, 50(3): 33.

Al Hamad, H. A., & Abu Zitar, R. 2010. Development of an efficient neural-based segmentation technique for Arabic handwriting recognition. *Pattern Recognition*, 43(8): 2773-2798.

Al Hamad, H. A., & Hamdi-Cherif, A. 2012. The Arabic Center for Document Analysis and Recognition (ACDAR)-Structure and Perspectives. *European Conference of Computer Science (ECCS'12)*, pp. 85-91.

Alaei, Alireza & Nagabhushan, P & Pal, Umapada. 2010. A Baseline Dependent Approach for Persian Handwritten Character Segmentation. Proceedings. *International Conference on Pattern Recognition (ICPR)*, pp. 1977-1980.

Alginahi, Y. M. 2013. A survey on Arabic character segmentation. *International Journal on Document Analysis and Recognition (IJDAR)*, 16(2): 105-126.

Amin, A., Masini, G., & Haton, J. 1984. Recognition of handwritten Arabic words and sentences. *7th International Conference on Pattern Recognition*, pp.1055-1057.

Bataineh, B., Abdullah, S. N. H. S., & Omar, K. 2011. An adaptive local binarization method for document images based on a novel thresholding method and dynamic windows. *Pattern Recognition Letters*, 32(14): 1805-1813.

Bataineh, B., Abdullah, S. N. H. S., & Omar, K. 2012. A novel statistical feature extraction method for textual images: Optical font recognition. *Expert Systems with Applications*, 39(5): 5470-5477.

Blumenstein, M., Liu, X. Y., & Verma, B. 2007. An investigation of the modified direction feature for cursive character recognition. *Pattern Recognition*, 40(2): 376-388.

Chahi, A., El khadiri, I., El merabet, Y., Ruichek, Y., & Touahni, R. 2018. Block wise local binary count for off-Line text-independent writer identification. *Expert Systems with Applications*, 93: 1-14.

Elaiwat, S., & Abu-Zanona, M. A.-A. 2012. A three stages segmentation model for a higher accurate off-line arabic handwriting recognition. *World of Computer Science and Information Technology Journal*, 2(3): 98-104.

Elnagar, A., & Bentrcia, R. 2012. A Multi-Agent Approach to Arabic Handwritten Text Segmentation. *Journal of Intelligent Learning Systems and Applications*, 4(03): 207.

Elnagar, A., & Bentrcia, R. 2015. A Recognition-Based Approach to Segmenting Arabic Handwritten Text. Journal *of Intelligent Learning Systems and Applications*, 7(04): 93.

Eraqi, H. M., & Abdelazeem, S. 2012. A new Efficient Graphemes Segmentation Technique for Offline Arabic Handwriting. *International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp. 95-100.

Firdaus F. I., Khumaini A. and Utaminingrum, F. 2017. Arabic letter segmentation using modified connected component labeling," *2017 International Conference on Sustainable Information Engineering and Technology (SIET)*, Malang, 2017, pp. 392-397.

Ghaleb, H., Nagabhushan, P., & Pal, U. 2017. Segmentation of offline handwritten Arabic text. Paper presented at the *1st International Workshop on Arabic Script Analysis and Recognition (ASAR)*. 3-5 April.

Gordon, R. G., & Grimes, B. F. 2005. *Ethnologue: Languages of the world*. SIL International,15: [online]/ Available at: http://www.ethnologue.com/15. [27 Jun 2018]

Khorsheed, M. S. 2002. Off-line Arabic character recognition–a review. *Pattern analysis & applications*, 5(1): 31-45.

Lawgali, A., Bouridane, A., Angelova, M., & Ghassemlooy, Z. 2011. Automatic segmentation for Arabic characters in handwriting documents. *18th IEEE International Conference on Image Processing (ICIP)*, pp. 3529-3532.

Nazif, A. 1975. A system for the recognition of the printed Arabic characters. Master's Thesis (2nd Edition). Faculty of Engineering, Cairo University.

Nidal Lamghari FSTG, L., Charaf, M. E. H., & Said Raghay FSTG, L. 2016. Template matching for recognition of handwritten Arabic characters using structural characteristics and Freeman code. *International Journal of Computer Science and Information Security*, 14(12): 31.

Parhami, B., & Taraghi, M. 1981. Automatic recognition of printed Farsi texts. *Pattern Recognition*, 14(1-6): 395-403.

Parvez, M. T., & Mahmoud, S. A. 2013. Offline Arabic handwritten text recognition: a survey. *ACM Computing Surveys (CSUR)*, 45(2): 23.

Radhiah, A., Machbub, C., Hidayat, E. M. I., & Prihatmanto, A. S. 2018. Printed Arabic letter recognition based on image. *In Signals and Systems (ICSigSys), 2018 International Conference on, IEEE*, pp. 86-91.

Ramdan, J., Omar, K., & Faidzul, M. 2017. A Novel Method to Detect Segmentation points of Arabic Words using Peaks and Neural Network. *International Journal on Advanced Science, Engineering and Information Technology*, 7(2): 625-631.

Stockman, G., & Shapiro, L. G. 2001. *Computer Vision*. 1st: Upper Saddle River, Nj: Prentice Hall, 2001-02-02.

Wshah, S., Shi, Z., & Govindaraju, V. 2009. Segmentation of Arabic handwriting based on both contour and skeleton segmentation. *10th International Conference on Document Analysis and Recognition (ICDAR'09)*, pp. 793-797.

Zidouri, A. 2006. Oran system: A basis for an arabic ocr. *Arabian Journal for Science and Engineering*, 31(1B): 91.

Dr. Tarik Abdel-Kareem Abu-Ain
College of Computing and Informatics
Saudi Electronic University, Saudi Arabia
t.aboain@seu.edu.sa

Assoc. Prof. Dr. Siti Norul Huda Sheikh Abdullah (Corresponding author)
Siti Zaharah Abd Rahman
Center for Cyber Security,
Faculty of Information Sciences and Technology,
Universiti Kebangsaan Malaysia,
snhsabdullah@ukm.edu.my,

Prof. Dr. Khairuddin Omar
Center For Artificial Intelligence Technology,
Faculty of Information Sciences and Technology,
Universiti Kebangsaan Malaysia.

Received: 13 August 2018
Accepted: 12 December 2019
Published: 30 June 2019