applied sciences

MDPI

Article

A New Computational Method for Arabic Calligraphy Style Representation and Classification

Zineb Kaoudja [1,*], Mohammed Lamine Kherfi [1,2] and Belal Khaldi [3]

[1] Lab or Artificial Intelligence and Data Science, Kasdi Merbah Ouargla University, Ouargla BP. 511, Algeria; kherfi.lamine@univ-ouargla.dz
[2] LAMIA Laboratory, University du Québec à Trois-Rivières, Trois-Rivières, QC G8Z 4M3, Canada
[3] LINATI Laboratory, University of Ouargla-Kasdi Merbah, Ouargla BP. 511, Algeria; bilel.khaldi@gmail.com
* Correspondence: zinebkaoudja@gmail.com

Abstract: Despite the importance of recognizing Arabic calligraphy styles and their potential usefulness for many applications, a very limited number of Arabic calligraphy style recognition works have been established. Thus, we propose a new computational tool for Arabic calligraphy style recognition (ACSR). The present work aims to identify Arabic calligraphy style (ACS) from images where text images are captured by different tools from different resources. To this end, we were inspired by the indices used by human experts to distinguish different calligraphy styles. These indices were transformed into a descriptor that defines, for each calligraphy style, a set of specific features. Three scenarios have been considered in the experimental part to prove the effectiveness of the proposed tool. The results confirmed the outperformance of both individual and combine features coded by our descriptor. The proposed work demonstrated outstanding performance, even with few training samples, compared to other related works for Arabic calligraphy recognition.

Keywords: document image analyzing; Arabic calligraphy; feature extraction; machine learning

check for updates

Citation: Kaoudja, Z.; Kherfi, M.L.; Khaldi, B. A New Computational Method for Arabic Calligraphy Style Representation and Classification. *Appl. Sci.* **2021**, *11*, 4852. https://doi.org/10.3390/app11114852

Academic Editor: Antonio Fernández-Caballero

Received: 16 April 2021
Accepted: 21 May 2021
Published: 25 May 2021

1. Introduction

Document Analysis (DA) is one of the computer vision applications. It is a discipline that combines image processing and machine learning techniques to process and extract information from document images of different text types such as digits or alphabet. DA attempts to extract the layout from a scanned document page and then reuse it to generate a new visually similar document with other content. Therefore, it must be capable of identifying writing characteristics, such as the style used in writing, and employ it to reproduce similar documents with the same style.

DA techniques have been employed in many applications such as document layout analysis [1], signature verification [2], writer identification [3], optical character recognition (OCR), optical Font recognition (OFR), etc. For instance, OFR aims to define the text writing style from a document image. Based on the purpose of the writing and the source of the written text, text writing can be classified into two categories, as shown in Figure 1. The first one is texts written by humans, which in turn comprises two sub-categories namely ordinary handwriting and calligraphy. The second one is machine-printed texts that may be written using common styles. Thus, text produced by machine or handwritten calligraphy uses well-known styles. Writing styles used by a machine are referred to as fonts, whereas calligraphy refers to the set of styles that might be used in handwriting texts. Fonts and calligraphy also differ in the purpose of use, calligraphy is mostly used for decoration such as in mosque walls, or in artists' paintings, whereas fonts are mostly used for documentation and communication. Fonts are far less complicated than calligraphy. They may provide alternatives, ligatures, and some stylized versions, but fundamentally, they can only be as creative as type designers. When writing letters by hand, there are no limitations to how to write them. Every version of a letter can be written differently. Letters may be merged or connected even though they are not adjacent. Consequently, fonts do not offer flexibility compared to handwriting. Such flexibility makes the recognition of calligraphy or reproduction of text using similar calligraphy styles an extremely difficult task for a DA system. In this work, we aim to propose a tool that is able to classify Arabic calligraphy styles.

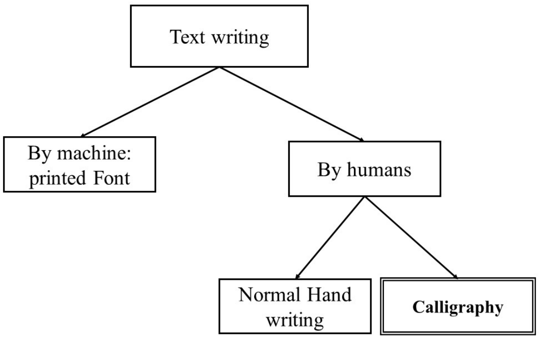


Figure 1. The different categories of text writing styles. Our work falls in the highlighted category of calligraphy text.

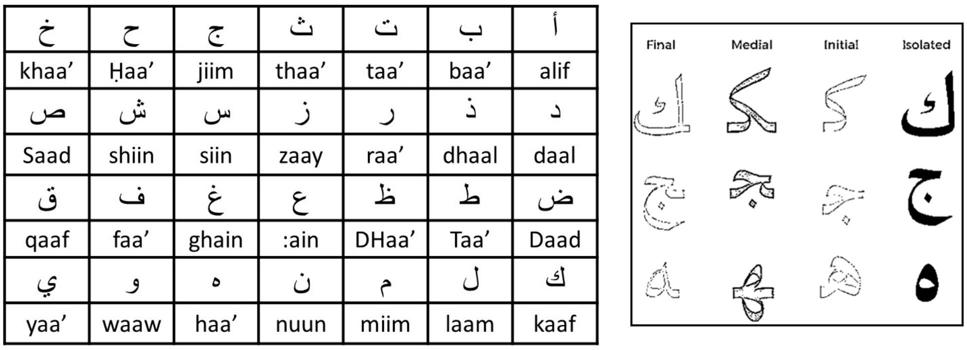
Calligraphy style recognition (CSR) is considered an important research area for the following reasons: (1) It helps to read the text content. In other words, knowing the used style reflects the rules used in the writing of different characters, which in turn eases the reading task. (2) It helps to recognize the different parts of a document. Some styles are defined to represent specific parts in the document such as titles, footers, paragraphs, etc. (3) It also helps to grasp the history of a document. In paleography, CSR is used to define the era in which the document was written because styles appeared in different eras. (4) It also helps to define the origins (i.e., geographical area) in which a document was written. (5) It can be used for tutor purposes. Calligraphy learners use CSR systems to judge the quality of their writings in cases of experts' absence.

Arabic is the fourth most spoken language in the world, and it is the first language of more than 200 million people across the world. Arabic script is the third most widely used script after Latin script [4]. Similar to other languages, it has grammar, spelling, punctuation rules, pronunciation, slang, and idioms. Several characteristics beyond the mere differences between languages make Arabic distinctive, including the number of variations and the written form. Arabic handwritten text is divided into two main parts: artistic writing called calligraphy and non-artistic (handwritten or printed) scripts. Arabic calligraphy (AC) is one of the most significant arts in the world. It is written only by hand. The first style of calligraphy was developed at the end of the 7th century. Over time, the Muslim and Arab world developed many other styles of calligraphy. Due to the complex shape of the Arabic text, one must hold a certain level of expertise to be able to write AC texts or recognize a writing style.

Before delving into Arabic calligraphy systems, we must shed some light on Arabic font recognition (AOFR) and how it differs from calligraphy recognition. On the one hand, the AOFR domain has received a lot of attention for several reasons: (a) the availability of public big datasets, (b) the ease of dealing with Arabic fonts due to their fixed shapes, (c) and the considerable amount of related works [5,6] on the subject. On other hand, far less attention has been given to Arabic calligraphy style recognition (ACSR). However, for the other languages, several works have been made to this end such as the work [7] in which topological features have been proposed to recognize Hebrew handwriting styles. Nonetheless, the complexity of calligraphy styles makes it possible to use only two characters, the character Aleph and the character Lamed. Other attempts have been carried out for Chinese style recognition in [8], proposing a feature extraction method based on the regional guided filter (RGF) with reference images, which is generated by K-Nearest

Neighbor (KNN) matting and used as the input image for RGF, and in [9] using deep features with Support Vector Machine (SVM) and Neural Network (NN) classifiers.

Despite the labor in other languages to find a common solution for style recognition, these features are not efficient and applicable for Arabic due to their nature. Arabic calligraphy texts are extremely smooth, interconnected, and have a unique characteristic in their form. Arabic has 28 letters; each one has isolated and connected forms (at the beginning, center, and end), as illustrated in Figure 2. Another difficulty in Arabic calligraphy is the possibility of letters overlapping one above the other, which is not the matter with other languages. Additionally, the shapes of letters are affected by both the writing and the writer's style, and also by the purpose of the writing itself (e.g., for decoration, documentation, or other purposes).



خ	ح	ج	ث	ت	ب	أ
khaa'	Ḥaa'	jiim	thaa'	taa'	baa'	alif
ص	ش	س	ز	ر	ذ	د
Saad	shiin	siin	zaay	raa'	dhaal	daal
ق	ف	غ	ع	ظ	ط	ض
qaaf	faa'	ghain	:ain	DHaa'	Taa'	Daad
ي	و	ه	ن	م	ل	ك
yaa'	waaw	haa'	nuun	miim	laam	kaaf

(a) (b)

Figure 2. Components of the Arabic language. (**a**) The alphabet consisting of 28 characters and (**b**) the different forms of the same character based on the position in the word.

In this paper, we propose a robust solution for AC style recognition from images. Our proposal consists of two main phases: (1) feature extraction from AC images and (2) classification through decision fusion. The extracted features, which represent the different morphologies of AC, have been used for the first time in this work and have been inspired by the characteristics experts (calligraphers) use for recognizing AC styles. Since each feature is dedicated to distinguishing one or more styles, multiple classifiers have been engaged in the process of recognition in which the decisions are fused to obtain a final decision. It should be mentioned that none of the former works on AC recognition has implicated such a wide variety of styles (nine styles) as we did in this work (Table 2). Our proposal achieved state-of-the-art and also provided stability against scale variation and learning from a few samples.

The remainder of this paper is organized as follows. In Section 2, we clarify Arabic writing characteristics, styles, the writing complexity of each style, and the reasons we consider nine styles. Section 3 lists and discusses highly related works in the field of AC style Recognition. Thereafter, we introduce our proposal in Section 4 and experimentally evaluate it in Section 5. Finally, we infer a general conclusion and some perspectives.

2. Characteristics of Arabic Handwritten Calligraphy and Its Complexity

Arabic consists of 17 main character forms. With the addition of dots placed above or below certain of them, it provides a total of 28 letters. As shown in (Figure 2) above, the same letter shape can form a "b" or "baa" sound when one dot is placed below ((ب)), a "t" or "taa" sound when two dots are placed above ((ت)), or a "th" or "thaa" sound when three dots are added above ((ث)). Short vowels are not included in the alphabet but instead indicated by signs (diacritics) placed above or below the consonant or long vowel that they follow. The Arabic writing system is a Right to Left (RTL), which means words are written and read in RTL way as in Hebrew, Persian, and Urdu. Texts are written in a cursive way where letters must be interconnected unlike other languages, such as English or French, where connecting letters is a matter of choice rather than obligation. One must also know that a letter in Arabic can be written in three different forms based on its location or syllable in the word (beginning, middle, or at the end) as shown in Figure 2.

Arabic Calligraphy Styles

AC consists of two main styles (Kufi and Naskh), each style has several variants, as well as region-specific styles. In the history of Arabic calligraphy, there were six main styles namely Kufi, Diwani, Farisi, Naskh, Rekaa, and Thuluth each of which has its characteristics and writing rules. Over time, over 100 different styles appeared and identified driven from the former main styles. Nine distinctive categories of cursive styles evolved, becoming ever more refined and elegant, some of which are still used and some others that have disappeared. The commonly used styles nowadays are the six basic styles (namely: Kufi, Diwani, Farisi, Naskh, Rekaa, and Thuluth) and three other styles that are driven from them which are Maghribi, Mohakik, and Square-Kufi. Unlike other related works that dealt with a few of these styles, in this work, we consider the full list of the nine styles. Table 1 resumes the different characteristics and the cues that experts use in identifying each one of these styles. It also provides a representative sample from each style to show how it differs from others.

Table 1. The nine most common AC styles with their characteristics and cues that are used by experts to identify them.

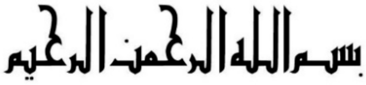
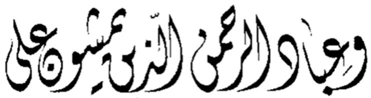
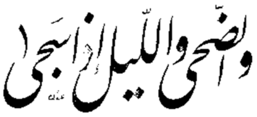
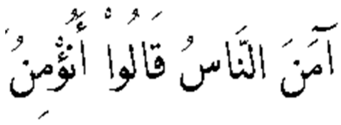
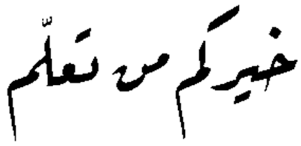
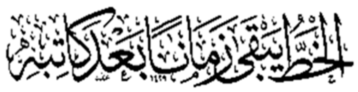
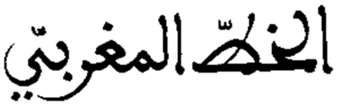
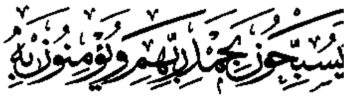
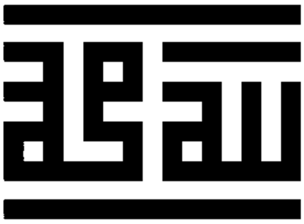
AC Style	Example Image	General Characteristics	Cues Used by Experts	Style Source
Kufi		Composed of geometrical forms such as straight verticals/horizontals lines and distinguishable angles.	- it is angular - thick strokes - long vertical extended strokes	
Diwani		A cursive script, beautiful, and harmonious. It is characterized by the rounded shape of letters. Words are written skewed descend from right to left. Lines start thick and then attenuate.	- slant words - written with a rounded shape - does not follow a straight line - the style width changes from thick to thin	
Farisi		Characterized by the letter's accuracy and extension, and by its ease and clarity and lack of complexity. Farisi (alt., Nasta'liq) is a cursive text.	- simple orientation - does not follow a straight line	
Naskh		One of the simplest writing types of calligraphy. Its name means "the copy style" because it has been used for copying books. It is recognizable by its balance and its plain clear forms. Nowadays, this script is used primarily in print.	-simple orientation - written on the baseline - slight change in the text thickness	
Rekaa		Mostly, used without diacritics. known with its clipped letters. Composed of short, straight lines and simple curves, as well as its straight and even lines. It is written above the baseline except for some specific letters.	- simple orientation - written above the baseline - thick stroke	
Thuluth		It is an elegant, cursive script that has large, elongated, and elegant letters. It has certain roles, but it could be changed according to calligraphers' needs. It has big letters, which take a big space above the baseline; the big overlapped curves result in big holes that calligraphers fill with artistic diacritics.	- overlapping letters - big rounded letters - high number of diacritics - it does not consider the baseline	
Maghribi		It has special letterforms which provide it with a unique beauty and make it easy to read, even in long texts. It is marked by descending lines written with very large bowls.	- text orientation - written on the baseline	-Kufi

Table 1. *Cont.*

AC Style	Example Image	General Characteristics	Cues Used by Experts	Style Source
Mohakik		Mohakik and Thuluth have almost the same characteristics. However, letters in Mohakik are longer under the baseline.	- special orientation under the baseline - same diacritics as the thuluth style - written on the baseline	-Thuluth
Square-Kufi		A unique style developed for decorating walls rather than papers. usually designed with mosaic faience, decorative glazed facing tiles, or simple bricks, rather than reed pens and ink. It has two strict rules: (a) evenness of full and empty spaces, and (b) square angles and strict lines.	- strokes with equal thickness - square angles	-Kufi

Due to the complexity of AC and the shared characteristics among its styles, none of the former related works has considered the full list of nine styles. However, there were some serious efforts to come up with some specific techniques for recognizing AC styles. In the following section, we list and discuss works that aimed at recognizing Arabic calligraphy styles using computer vision techniques.

3. Related Work

Arabic text font recognition has been extensively studied owing to the availability of a huge amount of resources that help researchers build new solutions. Works on Arabic font recognition can be grouped into two categories, namely handcrafted- and deep learning-based categories. On the one hand, starting with handcrafted-based solutions, Faten Kellal [10] uses discrete curvelet transform (DCT) to convert images into descriptors, and then, a backpropagation neural network has been employed to predict the style of the image text based on the extracted descriptor. In [11,12], the former method has been reinforced against scale changes by using a steerable pyramid texture. Hussein. A.K [5] has experimented and compared several texture descriptors for Arabic font recognition using extreme learning machine (ELM) and fast learning machine (FLN). On the other hand, deep convolutional neural networks (DCNN) [13,14] have been utilized in several occasions to perform Arabic font and font size recognition.

While there has been much research on Arabic font, few researchers have dealt with Arabic calligraphy. It could be attributed to the lack of resources such as public datasets and insufficiently related works. However, the reader should know that dealing with calligraphy is much harder than fonts due to the absence of creativity in the latter. In other words, calligraphy texts are affected by the calligrapher's impression, which is not the case with texts generated using machine fonts. Therefore, we limit ourselves to works that tackle the issue of recognizing calligraphy styles rather than fonts. Table 2 lists works conducted on AC style recognition with their respective datasets and the number of styles taken into account.

In [15], the authors argued that Arabic calligraphy images should be dealt with as textures, and therefore, they evaluated a set of commonly used texture descriptors for the task of AC identification. Results indicated that Binarized Statistical Image Features (BSIF) is the best texture descriptor among all for such a task.

Due to the lack of big datasets of AC and the fine-grained styles (e.g., Mohakik and Thuluth), most works in the literature have utilized classical machine learning tools instead of deep learning. Moreover, they attempted to propose their descriptors for this issue instead of using other common descriptors such as Sift, LBP, etc.

Table 2. List of related works on ACSR.

Author (Year)	Language	Dataset	N° Styles
Z.Kaoudja et al. (2020) [15]	Arabic	1685 line image	9
Batainah et al. (2012) [16]	Arabic jawi	700 blocks image (privet)	6
Batainah et al. (2013) [17]	Arabic-jawi	700 block image (privet)	7
Batainah et al. (2011) [18]	Arabic	14 documents images (privet)	Unknown
Batainah et al. (2011) [19]	Arabic	100 line image (privet)	6
Talab et al.(2011) [20]	Arabic	700 line images	7
Azmi et al. (2011) [21]	Arabic-jawi	100 character image (privet)	5
Azmi et al. (2011) [22]	Arabic-jawi	1019 character images (privet)	4
Adam et al. (2017) [23]	Arabic	330 character images (privet)	6
Allaf et al.(2016) [24]	Arabic	267 line/word images (privet)	3
Elhmouz et al. (2020) [25]	Arabic	421 line/word images (privet)	3
M. Khayyat (2020) [26]	Arabic	2653 documents images (privet)	6
Z.Kaoudja et al. (2019) [27]	Arabic	1685 line image	9
R.F. Moghaddam et al. (2012) [28]	Arabic	27709 word image	Unknown

In all of their works, Bilel Bataineh [16–19] and Ahmed Talab [20] attempt to classify AC styles with a proposed statistical descriptor, named Edge Direction Matrix (EDMs). They firstly pass through two steps called EDM1 and EDM2, which produce 3 × 3 matrices counting edges (i.e., pixels' adjacency), and then extract 22 statistical moments to constitute the final image descriptor. Unlike the aforementioned global approach, the works in [21–23] focused on extracting features using local shapes of letters. For instance, Allaf [24] uses the text geometrical shape, the density of text above and under the baseline, number of diacritics up and down the text, text orientation, the position of the text baseline, and the ratio of black and white for the whole image. However, the validity of this solution has been proven using a small dataset (260 images) comprising only three styles, which is highly insufficient. To sum up, the aforementioned works can be classified into two categories, holistic and local approaches. The holistic one is concerned with recognizing styles using global descriptors. However, these approaches are not dedicated to calligraphy, but they can be applied to heterogeneous images ignoring the very specific characteristics of AC. On the other hand, the local approaches look into the characteristics of AC, exaggeratedly, via letter segmentation which makes it inapplicable especially for complicated styles or for big datasets. In this paper, we propose a solution that combines the advantages of both, describing the specific features of each style without resorting to letter segmentation. Although deep learning has proven itself as the best solution for many issues, it has not been widely explored for ACSR due to the fine-grained styles and absence of big datasets. We list the only two works that utilize deep learning nets for ACSR, which are [25,26]. In the former, a common deep model (stacked-auto encoder) has been trained and utilized for feature extraction, and a SoftMax has been employed to perform the recognition process. In the latter, the full MobileNetV1 model has been fine-tuned and used to classify six AC styles.

4. Proposed Method

In this paper, a new tool is proposed for the task of ACSR. The proposal is capable of recognizing nine different AC styles and might be extended to other styles. After analyzing Arabic calligraphy, we found out that the styles are geometrically different. Some styles have a unique shape, some share similar letter shapes, and some others have special diacritics. For the visually different styles, it is easy to separate them, whilst for the other fine-grained styles, we needed to extract one or more distinctive characteristics to separate them. The proposed solution was inspired by the calligraphy expert. A calligraphy expert uses tools such as brushes, pens, or markers to create a special writing style that is artistic and expressive. After an in-depth study of each style, looking for the visual characteristics used by calligraphers to distinguish it, we did come up with the main distinctive features that need to be extracted for each style or set of styles. In the feature extraction section, we explain these visual features and the relationship with their corresponding styles.

Our proposed tool consists of three main steps: (a) image preprocessing: perform several transformations to prepare the image for feature extraction, (b) feature extraction: extract the proposed descriptor from each image, and (c) classification: use the extracted features to predict the style of the input image. Figure 3 illustrates the general scheme of our proposed tool.

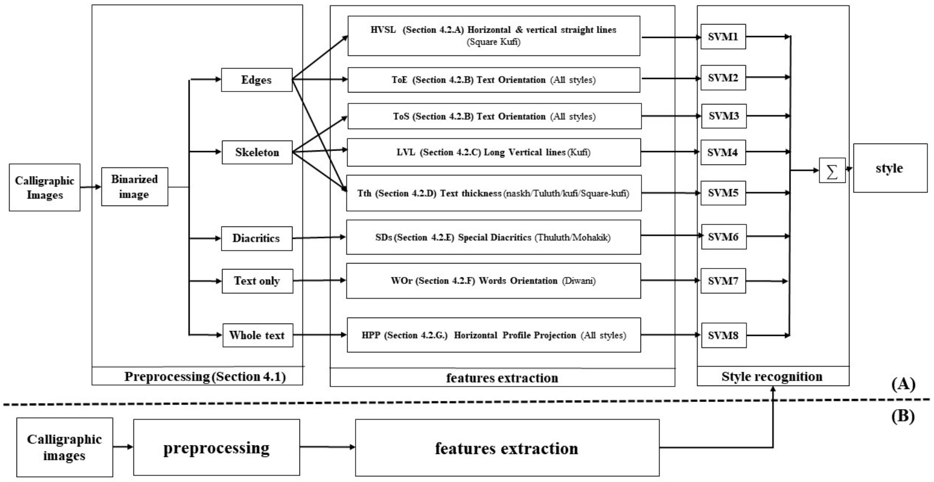

Calligraphic Images
Binarized image
Edges
Skeleton
Diacritics
Text only
Whole text
Preprocessing (Section 4.1)
HVSL (Section 4.2.A) Horizontal & vertical straight lines (Square Kufi)
ToE (Section 4.2.B) Text Orientation (All styles)
ToS (Section 4.2.B) Text Orientation (All styles)
LVL (Section 4.2.C) Long Vertical lines (Kufi)
Tth (Section 4.2.D) Text thickness (naskh/Tuluth/kufi/Square-kufi)
SDs (Section 4.2.E) Special Diacritics (Thuluth/Mohakik)
WOr (Section 4.2.F) Words Orientation (Diwani)
HPP (Section 4.2.G.) Horizontal Profile Projection (All styles)
features extraction
SVM1
SVM2
SVM3
SVM4
SVM5
SVM6
SVM7
SVM8
Style recognition
∑
style
(A)
(B)
Calligraphic images
preprocessing
features extraction

Figure 3. A general schema for our proposed tool. The dotted line separates the (**A**) training from (**B**) test phases. The images pass through a processing sequence starting from preprocessing, features extraction, and finally classification. The final descriptor contains eight features, as shown, which will be explained in Section 4.2.

4.1. Preprocessing

Since we are interested only in the morphology of text letters, all images are first converted into binary (i.e., black text on a white background) after having the texts separated from backgrounds [29]. It should be known that most images contain either meaningful texture, which is a part of the decoration or a meaningless one that resulted from the noise while capturing. In either case, we illuminate the background so it does not affect the results. Thereafter, we apply the following micro-operations to extract, for each image, a set of images namely edge image, skeleton image, diacritics image, and no-diacritics image. Each of these generated images will thereafter be used to define the characteristics of one or more styles of AC. Figure 4 shows some of the images that resulted from these operations.

4.2. Feature Extraction

In most fields, it is a common practice to ask experts about cues they use to decide for a certain inquiry. Calligraphers use some specific features, which we will discuss hereafter, to find the appropriate style for a given text. The main aim of this step is to transform these features into values that can be fed to a machine so it can decide on the style of a given text image. It should be mentioned that a feature might specify one or more styles. Features designated for each style or set of styles might be used in a sequential manner (sequential decision) or parallel manner. However, in our work, we considered all the features to be equally important and, therefore, adopt a parallel manner. That is to say, each feature descriptor is fed to a classification machine, SVM in our case, and the final decision is the combination of all the decisions of these machines.

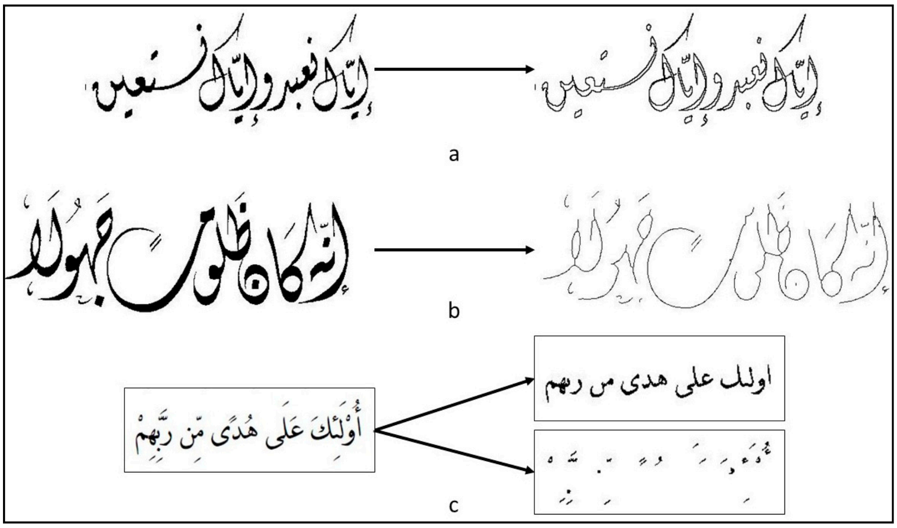


Figure 4. Operations executed in the preprocessing step to generate a set of accompanying images for each image in the dataset. Each generated image will thereafter be used to extract one or a set of features. These operations are (**a**) edge detection using a 3 × 3 Laplacian filter, (**b**) skeleton detection using 'Zhang-Suen Thinning Algorithm' [30], and (**c**) separating diacritics from text using the flood-fill algorithm [31].

4.2.1. Horizontal and Vertical Straight Lines (HVSL)

This descriptor has been designed to separate Square-Kufi from other styles. It mainly describes how frequently horizontal and vertical lines appear in a text image. Square-Kufi differs from other styles by the high number of vertical and horizontal lines as shown in (Figure 5).

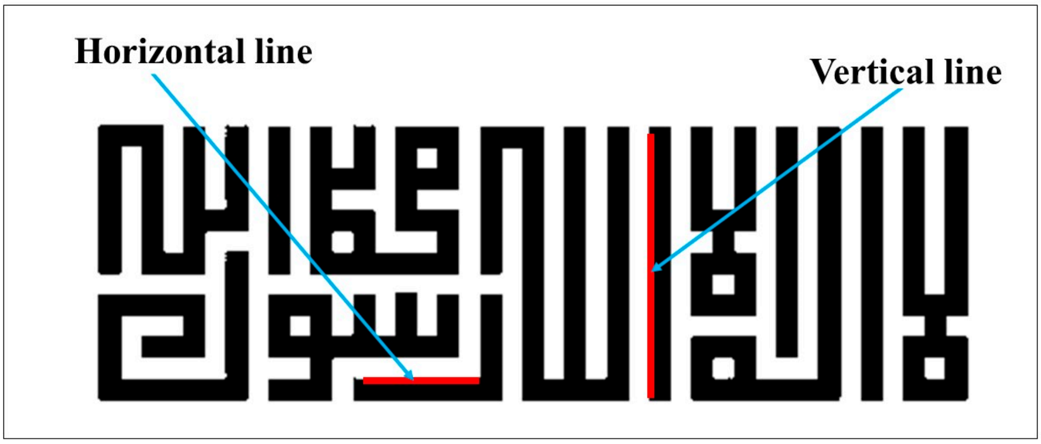


Figure 5. Vertical and horizontal lines as basic features of Square-Kufi.

HVSL is not extracted from the input image, but rather from the edge image resulted from the preprocessing step. The final HVSL descriptor holds the following features: (a) The appearance frequency of vertical and horizontal (i.e., V/H) lines, which is the main cue used to distinguish Square-Kufi, (b) The difference ratio between the number of the pixels that constitute the texts edge and the sum of V/H lines appearance frequency.

4.2.2. Text Orientation from Edge/Skeleton (ToE/ToS)

One of the main characteristics of all the Arabic calligraphy styles is text orientation. Each style is usually written in a specific direction that defines its unique shape. By direction, we mean the slope of the pen while writing words. For instance, the pen in Kufi is used to write on the baseline without any slope, whereas, in Rekaa, the writing is slightly sloping. The pen itself might be held by the calligrapher vertically or in a sloppily way based on the style intended to be adopted. A vertical grasp of the pen results in flat tails whereas a sloping grasp results in pointed ones, as Figure 6 shows.

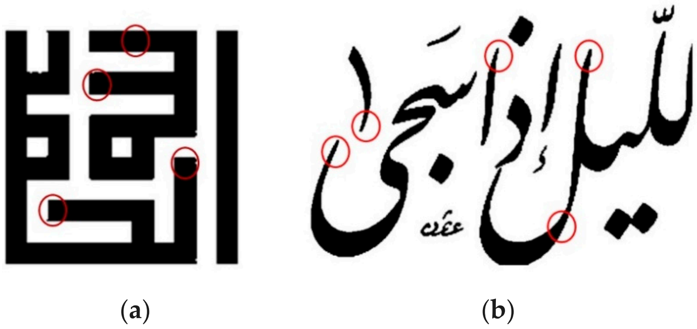


Figure 6. The effect of pen hold manner on the writing. In (**a**), the pen is held straight vertical whereas, in (**b**), the pen is held in a sloping manner.

To capture the orientation of words, we utilize the skeleton image, whereas to capture the effect of pen sloping, we use the text edge image. From each image, the orientation at each pixel level is extracted and then codified into a histogram (i.e., ToE for edge orientations and ToS for skeleton orientations). Figure 7 shows a representative example of the process.

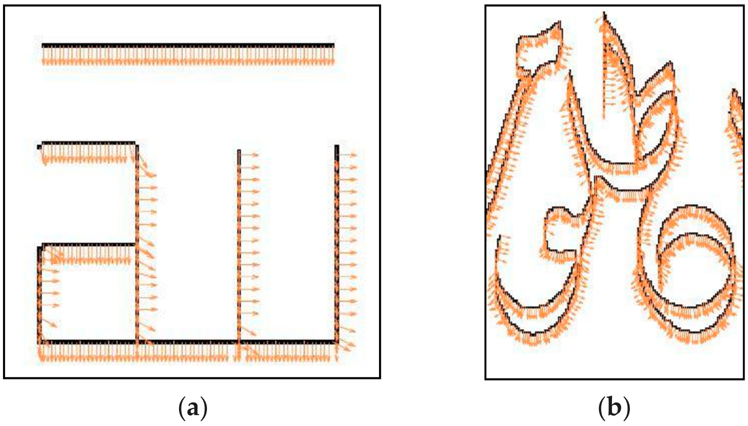


Figure 7. Extracting orientations to be used to generate ToE and ToS histograms. Orientations extracted from (**a**) the skeleton image, and (**b**) the edge image.

4.2.3. Long Vertical Lines (LVL)

Since Kufi and Square-Kufi share common features such as vertical and horizontal lines, HVSL will consider Kufi images as Square-Kufi and separate them from other styles. However, Kufi has a distinctive characteristic, compared to Square-Kufi, which is the long vertical straight lines. The Long Vertical Lines (LVL) descriptor has been designated to eliminate the conflict between texts written in kufi and the ones of Square-Kufi. It mainly describes how vertical lines in a text vary.

After having the vertical straight lines extracted from the skeleton image as Figure 8 shows, the following five measurements are calculated: (a) the text height from the bottom to top, (b) the number of detected vertical lines, (c) the length of the highest detected vertical line, (d) the difference ratio between the text height and the highest vertical line, and (e) the variance among the vertical lines.

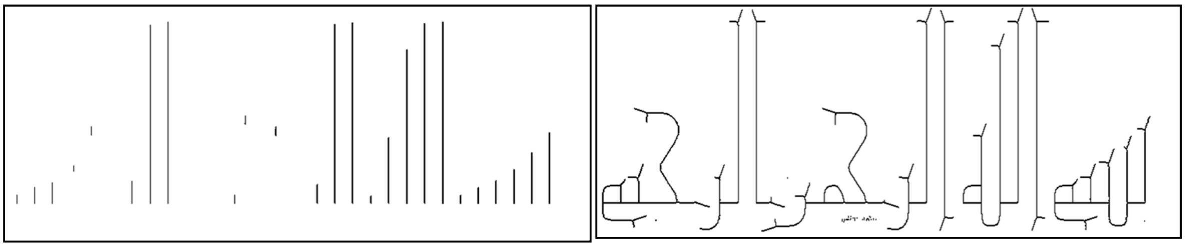


Figure 8. Vertical straight lines detection.

4.2.4. Text Thickness (Tth)

Stroke thickness plays an important role in defining the style. Some styles use a flat pen, whereas some others use a pointed one. In some styles, calligraphers alter the thickness while writing (via pushing down the pen or the opposite), whereas in others, the thickness is always preserved. Modeling such a feature in form of a descriptor will help the machine to understand more specificities of each style. Text thickness (Tth) descriptor codifies the appearance frequency of different line thicknesses in a text image. To extract this descriptor, we employ both the skeleton and edge image. To find the thickness level at a pixel of the skeleton, we calculate the distance between the two points on a perpendicular line that passes through p.

4.2.5. Special Diacritics (SDs)

Thuluth and Mohakik have a similar writing style that is decorated with diacritics having special shapes, as shown in Figure 9. An SDs descriptor will be used to inspect the existence of such diacritics in a given text image. To this end, each of the diacritics represented in Figure 9 is explicitly segmented and then represented with a vector of Hu moments [32]. Thereafter, the distance among these moments and those extracted from the input image's diacritics are calculated to decide whether the image is Thuluth/Mohakik or not.

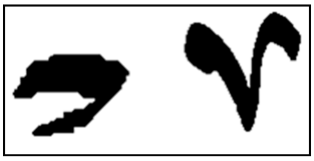


Figure 9. Thuluth and Mohakik diacritics.

It is worth mentioning that SDs has not been designed to identify Thuluth from Mohakik but rather to separate both of them from other styles.

4.2.6. Word's Orientation (WOr)

One of the salient features of the Diwani style is the words written in a slanted format, as Figure 10 shows. WOr descriptor mainly specifies how, on average, words in text images are oriented. The flood-fill algorithm is used to detect words from no diacritics images that are generated in the processing step. After calculating each word's orientation, the mean orientation along with the number of detected words are combined to form the final WOr descriptor.

WOr algorithm was used to distinguish Diwani from other styles. Diwani style yield an orientation average of about 45 degrees compared to 0 degrees by other styles.

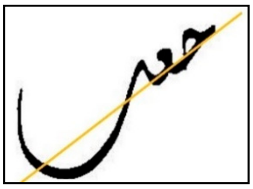


Figure 10. A slanted word from Diwani style, with its orientation measured, detected using the flood-fill algorithm.

4.2.7. Horizontal Profile Projection (HPP)

In some AC styles, all words are written on a baseline, whereas in some other styles, words are not subjected to a baseline. Horizontal profile projection is a descriptor that specifies how words are vertically spread within a text image. First, a sub-image that exactly fits the text is cropped from the text image, and then, all pixels are projected to a vertical histogram, using the following Equation (1), to find how pixels are vertically spread.

$$HPP = \sum_N I(M,N) / \max_M \left(\sum_N I(M,N)\right) \tag{1}$$

In an image in which words are written on a baseline, HPP will have only one bin with a high value, whilst, in an image in which words are vertically spread, HPP will have more than one bin with high value.

5. Experimental Results and Discussion

This section is dedicated to evaluating the performance of the proposed solution. Three scenarios have been carried out; the first one is meant to evaluate each descriptor individually to validate the role it has been proposed for in the first place. In the second scenario, the decisions yielded by all classifiers are combined to produce a final decision. We devote this final scenario to examine the effect of using few samples for training. All the experiments have been carried out under the following configuration:

- **Dataset:** We use the same dataset adopted in our previous works [15,27]. It consists of nine (9) classes each of which represent an Arabic style. The total number of the dataset's images is 1685. Each style has several images that vary from 180 to 195.
- **Classification:** For image classification, we use the well-known classifier Support Vector Machine (SVM). Support Vector Machine classifier is a supervised machine learning technique. The main goal of this supervised method is to find a function in a multidimensional space that is able to separate training data with known class labels. SVM can be used for linear as well as nonlinear data classification. In our case, we use the Polynomial Kernel. This choice is a result of the outperformance SVM showed compared to other classifiers. To avoid overfitting, 3-fold cross-validation has been adopted.
- **Metrics:** To validate our model, we use four metrics, namely: Recall (R), Precision (P), F1-score, and Accuracy given by the following formulas.

$$R = \frac{treue\ positif}{true\ positif + false\ negatif}$$

$$P = \frac{true\ positif}{true\ positif + false\ positif}$$

$$F1 - score = 2 \cdot \frac{R \cdot P}{R + P}$$

$$Accuracy = \frac{true\ positif + true\ negatif}{true\ positif + false\ positive + true\ negatif + false\ negatif}$$

Scenario 1. Experimenting with each descriptor individually:

As we discussed beforehand, some descriptors have been mainly proposed to identify one single style, whereas some others are general and can be used to separate multiple styles. This scenario is dedicated to analyzing the results for each descriptor individually. It should be mentioned that each descriptor has been used to train a separated SVM. Table 3 presents the accuracies yielded by each feature descriptor for each style.

Table 3. The accuracies yielded by each descriptor. The first column holds descriptors with the respective style they have been proposed for. Horizontal and Vertical Straight Lines (HVSL), Text orientation from Edges (ToE), Text orientation from Skeleton (ToS), Long Vertical Lines (LVL), Text thickness (Tth), Special Diacritics (SDs), Word's Orientation (WOr), Horizontal Profile Projection (HPP).

	Diwani	Naskh	Farisi	Rekaa	Thuluth	Maghribi	Kufi	Mohakik	Square-Kufi
HVSL (Square-Kufi)	63%	62%	34%	37%	83%	62%	87%	74%	100%
ToE (General)	93%	88%	94%	87%	88%	81%	94%	86%	99%
ToS (General)	96%	88%	91%	83%	90%	80%	91%	86%	98%
LVL (Kufi)	54%	77%	28%	44%	49%	34%	89%	25%	91%
Tth (General)	47%	61%	51%	52%	66%	51%	76%	43%	82%
SDs (Thuluth + Mohakik)	13%	2%	0%	10%	86%	11%	17%	62%	18%
WOr (Diwani)	85%	24%	9%	21%	9%	6%	8%	18%	62%
HPP (General)	76%	97%	58%	76%	83%	92%	97%	78%	100%

From Table 3, we can see that the descriptors that have been proposed for one or two specific styles have performed adequately. HVSL and WOr as instances generated accuracies of 100% and 85% respectively, which are the highest ones compared to others. However, LVL yielded a similar performance for both Kufi and Square-Kufi, although it has been proposed for the former. This could be attributed to the similarity between these two styles. The Tth descriptor works well for several styles that are characterized by their thickness. Those styles are Naskh, Thuluth, Kufi, and Square-Kufi. The highest accuracy was for Square-Kufi with a correct rate of 82%. In contrast, the lowest accuracy was for Mohakik style with 43%. HPP descriptor has worked well with most of the styles because it is a representation of a general characteristic. The lowest accuracy, which is 58%, produced with the style Farisi because it can be written in different manners (some texts are written on the baseline; in some others, words are written one above another).

We can conclude that general descriptors have the highest accuracies. Nevertheless, by combining these general descriptors with the specific ones, even better results can be achieved. In the following scenario, we evaluate the impact of combining all the descriptors and compare the results to other proposed methods including deep learning.

Scenario 2: combine the descriptors

After having proved the efficiency of the individual descriptors, we combine them to get a powerful descriptor that can be used to distinguish the nine AC styles effectively. To this end, each descriptor is used to train a separated classifier (SVM in our case) and the overall decision will be the sum of the decisions generated by all the classifiers. In other words, the final decision will be the one with the highest votes. Figure 11 represents the precision and recall yielded by our combined descriptor compared to other methods from state of the art [16,17,20,24,25].

From Figure 11, it seems that our proposed method outperforms all other related works with a precision of 97%. This remarkable performance of ours is a result of combining all AC features used by calligraphers to recognize different styles in one descriptor. The method proposed in [20] (EDM1+LBP) has scored the second-best performance. This is because the latter deals with AC images as texture that is an approach; we prove its effectiveness in [22]. Although the work [23] has opted for morphological descriptors as we did, the results were too weak. This is because the authors have taken into account the general features only and did not consider the specificity of each AC style. On the other hand, stacked auto-encoder has scored the worst precision (resp. recall) among all. This is because using stacked auto-encoders needs tens of thousands of images to tune thousands of parameters that constitute the network, especially since the input of the network is the pixels of the image itself rather than a descriptor. Yet, no AC dataset contains this huge number of images. To further evaluate the performance of these methods, F1-score at the style-level has been estimated and listed in Table 4.

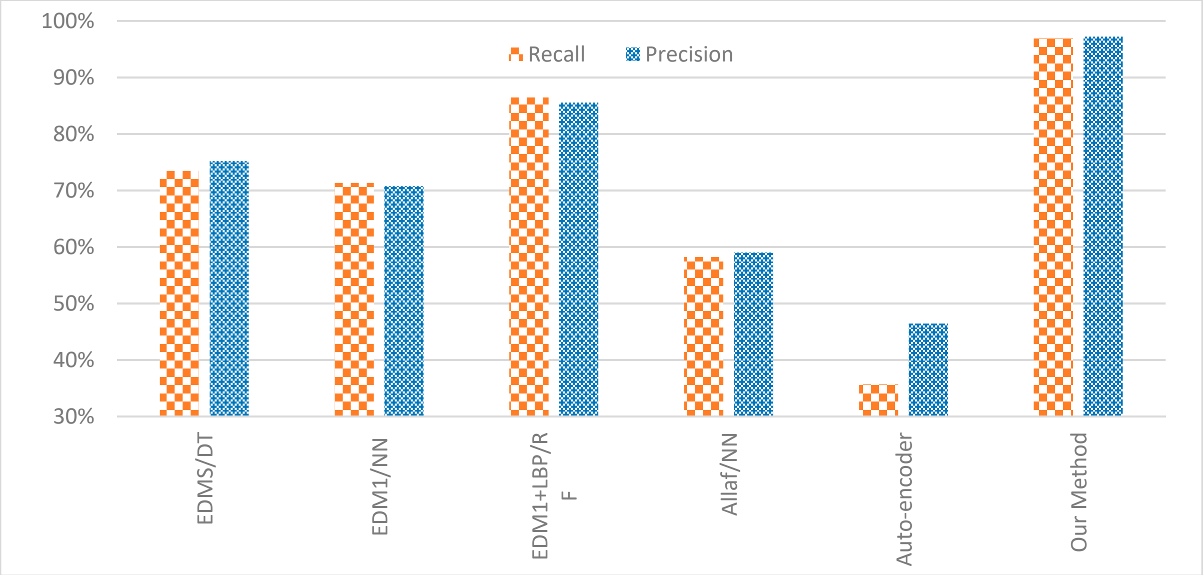


Figure 11. Recall and precision generated by the proposed method compared to other AC-related works.

Table 4. F1-score at style-level yielded by our method and other related works.

	EDMS/Decision Tree [16]	**EDM1/NN** [17]	**EDM1+LBP/Random Forest** [20]	**Allaf/NN** [24]	**Auto-Encoder** [25]	**Our Method**
Diwani	76%	75%	90%	30%	35%	98%
Naskh	92%	87%	98%	81%	48%	99%
Farisi	56%	58%	80%	23%	2%	97%
Rekaa	54%	53%	77%	45%	12%	98%
Thuluth	63%	56%	78%	73%	43%	94%
Maghribi	66%	77%	75%	46%	15%	97%
Kufi	64%	72%	93%	73%	14%	98%
Mohakik	41%	60%	79%	45%	33%	94%
S-Kufi	96%	94%	97%	94%	80%	99%

At first glance, Table 4 shows that our proposed method outperforms other methods in all styles. S-kufi seems to be the most distinguishable style among the others due to its unique features. In contrast, Thuluth, Mohakik, and Rakaa seem to be harder to distinguish. This can be attributed to the common way of writing of Rakaa and the similarity between Mohakik and Thuluth. Nevertheless, our proposed method has yielded satisfying results because we took advantage of slight changes among styles such as the unique way of writing diacritics. With most of the styles, our proposed method has produced more than 97% accuracy. However, with Thuluth and Mohakik, our method yields the lowest accuracy, which is 94%. To get an idea about how the styles Thuluth and Mohakik are misclassified, we generate and present the confusion matrix in Table 5.

Table 5. The confusion matrix, of the nine AC styles, generated by our proposed method.

Style	Diwani	Naskh	Farisi	Rekaa	Thuluth	Maghribi	Kufi	Mohakik	S-Kufi
Diwani	100%	0%	0%	0%	0%	0%	0%	0%	0%
Naskh	0%	100%	0%	0%	0%	0%	0%	0%	0%
Farisi	2%	0%	96%	2%	0%	0%	0%	0%	0%
Rekaa	0%	0%	1%	98%	0%	1%	0%	0%	0%
Thuluth	0%	0%	1%	0%	95%	0%	0%	3%	1%
Maghribi	0%	0%	1%	0%	1%	99%	0%	0%	0%
Kufi	0%	0%	0%	0%	0%	2%	98%	0%	0%
Mohakik	0%	0%	0%	0%	5%	0%	0%	94%	1%
S-Kufi	0%	0%	0%	1%	0%	0%	1%	0%	99%

From Table 5, we can see that our proposed method produces satisfying results with most of the styles. However, Mohakik and Thuluth have misclassified one as the other in some cases, which decreases the accuracy. This is because both styles Mohakik and Thuluth have similar shapes with less unique features that can be used to distinguish one from the other.

Scenario 3: Compare to other texture and deep learning methods

In a previous work [15], we proved that AC is better to be dealt with as texture. Since AC text is relatively a homogeneous (i.e., stationary) texture with repeated patterns (characters, diacritics, etc.), using statistical feature descriptors yields better results than learning-based including deep learning [33]. Furthermore, descriptors that are designed to be used with heterogenous images are harder to train (i.e., more images are needed) than descriptors dedicated to a specific family of images. That is to say, all the descriptors that we compare to our method will poorly perform in cases of a small training set. To prove these claims, an evaluation of statistical against learning-based (deep learning) methods in both normal and small training set size has been carried out. Figure 12 represents precision obtained in both cases.

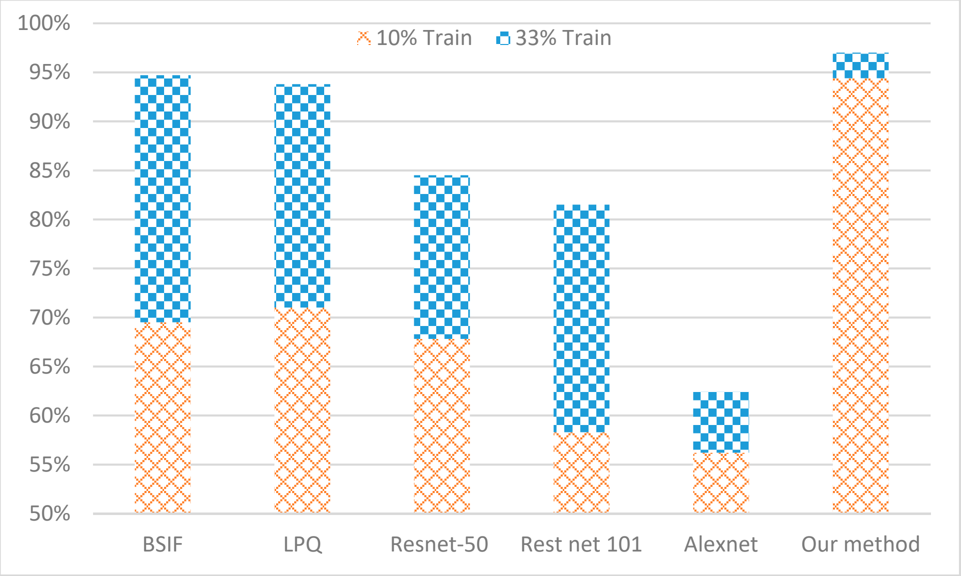


Figure 12. Related works evaluated and compared against our method using 33% and 10% images subset respectively for training.

From Figure 12, it appears that our method outperforms all other methods in both 33% and 10% training cases. As we have stated above, statistical descriptors (e.g., BSIF and LPQ) that are designed to be used with heterogeneous textures have performed well with AC style recognition. However, we have claimed that such descriptors lose performance in the case of using a few samples for training. The proof of this claim can easily be noted in the accuracy drop of all methods in the case of using a small image set for training. Our method, on the other hand, seems not to be greatly affected by reducing the size of the training subset to 10%. By this last scenario, we confirm that using dedicated descriptors for some specific datasets, which is the case with AC, is far more effective than general descriptors.

6. Conclusions

The main aim of this research was to investigate employing cues used by the calligrapher for AC style recognition. To this end, the features used by calligraphers have been codified into descriptors, some of which are dedicated to distinguishing specific styles and some others being for general features and some others are for general features. We did experiment our proposed descriptor in two different ways as individual features, as some descriptors have been made to identify one single style, whereas some others are general and can be used to separate multiple styles; in combined form, we combined them to get a powerful descriptor that can be used to distinguish the nine AC styles effectively using a rich dataset containing 1685 images categorized into nine styles. The results indicated the outperformance of the proposed method compared to other related works including deep learning. By these results, we have confirmed that exploiting the calligrapher's expertise rather than using general image features highly improves the performance of the AC recognition. Furthermore, the experiment outcomes proved that our descriptor was not highly affected by the small size of the training data, unlike other general-purpose descriptors. In this work, we have considered the most common features used by calligraphers in distinguishing styles. However, additional efforts should be spent to explore other less common features. Such features will surely further improve the results and extend the system's capability to other styles. Another aspect that needs to be tackled is text transformation, such as rotation, which is the case with most decorative texts.

Author Contributions: Conceptualization, Z.K., M.L.K. and B.K.; methodology, Z.K.; software, Z.K.; validation, Z.K., M.L.K. and B.K.; formal analysis, Z.K. and B.K.; investigation, Z.K.; resources, Z.K.; data curation, Z.K.; writing—original draft preparation, Z.K.; writing—review and editing, M.L.K. and B.K.; visualization, Z.K.; supervision, M.L.K.; project administration, M.L.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no potential conflict of interests.

References

1. Binmakhashen, G.M.; Mahmoud, S.A. Document Layout Analysis: A Comprehensive Survey. *ACM Comput. Surv.* **2019**, *52*, 109. [CrossRef]
2. Stauffer, M.; Maergner, P.; Fischer, A.; Riesen, K. A Survey of State of the Art Methods Employed in the Offline Signature Verification Process. In *New Trends in Business Information Systems and Technology: Digital Innovation and Digital Business Transformation*; Dornberger, R., Ed.; Springer Science and Business Media LLC: Berlin/Heidelberg, Germany, 2020; pp. 17–30.
3. Rehman, A.; Naz, S.; Razzak, M.I. Writer identification using machine learning approaches: A comprehensive review. *Multimed. Tools Appl.* **2019**, *78*, 10889–10931. [CrossRef]
4. Available online: https://www.worldatlas.com/articles/the-world-s-most-popular-writing-scripts.html (accessed on 10 May 2021).
5. Hussein, A.K. Fast learning neural network based on texture for Arabic calligraphy identification. *Indones. J. Electr. Eng. Comput. Sci.* **2021**, *21*, 1794–1799. [CrossRef]
6. Luqman, H.; Mahmoud, S.A.; Awaida, S. Arabic and Farsi Font Recognition: Survey. *Int. J. Pattern Recognit. Artif. Intell.* **2015**, *29*, 1553002. [CrossRef]

7. Bar Yosef, I.; Kedem, K.; Dinstein, I.; Beit-Arie, M.; Engel, E. Classification of hebrew calligraphic handwriting styles: Preliminary results. In Proceedings of the First International Workshop on Document Image Analysis for Libraries, Palo Alto, CA, USA, 23–24 January 2004; Institute of Electrical and Electronics Engineers (IEEE): Piscataway, NJ, USA, 2004; pp. 299–305.
8. Wang, L.; Gong, X.; Zhang, Y.; Xu, P.; Chen, X.; Fang, D.; Zheng, X.; Guo, J. Artistic features extraction from chinese calligraphy works via regional guided filter with reference image. *Multimed. Tools Appl.* **2018**, *77*, 2973–2990. [CrossRef]
9. Pengcheng, G.; Gang, G.; Jiangqin, W.; Baogang, W. Chinese calligraphic style representation for recognition. *Int. J. Doc. Anal. Recognit.* **2017**, *20*, 59–68. [CrossRef]
10. Kallel, F.; Mezghani, A.; Kanoun, S.; Kherallah, M. Arabic Font Recognition Based on Discret Curvelet Transform. In *Proceedings of the Third International Afro-European Conference for Industrial Advancement—AECIA 2016*; Advances in Intelligent Systems and Computing Series; Springer Science and Business Media LLC: Berlin/Heidelberg, Germany, 2018; Volume 565, pp. 360–369.
11. Jaiem, F.K.; Kherallah, M. A novel Arabic font recognition system based on texture feature and dynamic training. *Int. J. Intell. Syst. Technol. Appl.* **2017**, *16*, 289.
12. Jaiem, F.K.; Slimane, F.; Kherallah, M. Arabic font recognition system applied to different text entity level analysis. In Proceedings of the 2017 International Conference on Smart, Monitored and Controlled Cities (SM2C) 2017, Sfax, Tunisia, 17–19 February 2017; pp. 36–40.
13. Sakr, G.; Mhanna, A.; Demerjian, R. Convolution Neural Networks for Arabic Font Recognition. In Proceedings of the 2019 15th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), Sorento, Italy, 26–29 November 2019; Institute of Electrical and Electronics Engineers (IEEE): Piscataway, NJ, USA, 2019; pp. 128–133.
14. Amer, I.M.; Mostafa, M.G. Deep Arabic Font Family and Font Size Recognition. *Int. J. Comput. Appl.* **2017**, *176*, 1–6. [CrossRef]
15. Kaoudja, Z.; Khaldi, B.; Kherfi, M.L. Arabic Artistic Script Style Identification Using Texture Descriptors. In Proceedings of the 2020 1st International Conference on Communications, Control Systems and Signal Processing (CCSSP), El Oued, Algeria, 16–17 May 2020; Institute of Electrical and Electronics Engineers (IEEE): Piscataway, NJ, USA, 2020; pp. 113–118.
16. Bataineh, B.; Abdullah, S.N.H.S.; Omar, K. A novel statistical feature extraction method for textual images: Optical font recognition. *Expert Syst. Appl.* **2012**, *39*, 5470–5477. [CrossRef]
17. Bataineh, B.; Abdullah, S.N.H.S.; Omar, K.; Batayneh, A. Arabic-Jawi Scripts Font Recognition Using First-Order Edge Direction Matrix. In Proceedings of the International Multi-Conference on Artificial Intelligence Technology, Shah Alam, Malaysia, 28–29 August 2013; pp. 27–38.
18. Bataineh, B.; Abdullah, S.N.H.S.; Omar, K. Arabic calligraphy recognition based on binarization methods and de-graded images. In Proceedings of the 2011 International Conference on Pattern Analysis and Intelligent Robotics, ICPAIR 2011, Putrajaya, Malaysia, 28–29 June 2011; pp. 65–70.
19. Bataineh, B.; Abdullah, S.N.H.S.; Omar, K. Generating an Arabic Calligraphy Text Blocks for Global Texture Analysis. *Int. J. Adv. Sci. Eng. Inf. Technol.* **2011**, *1*, 150–155. [CrossRef]
20. Talab, M.A.; Abdullah, S.N.H.S.; Razalan, M.H.A. Edge direction matrixes-based local binary patterns descriptor for invariant pattern recognition. In Proceedings of the 2013 International Conference on Soft Computing and Pattern Recognition (SoCPaR), Hanoi, Vietnam, 15–18 December 2013; Institute of Electrical and Electronics Engineers (IEEE): Piscataway, NJ, USA, 2013; pp. 13–18.
21. Azmi, M.S.; Omar, K.; Nasrudin, M.F.; Ghazali, K.W.M.; Abdullah, A.; Abdullah, A. Arabic calligraphy identification for Digital Jawi Paleography using triangle blocks. In Proceedings of the 2011 International Conference on Electrical Engineering and Informatics, Bandung, Indonesia, 17–19 July 2011; IEEE: Piscataway, NJ, USA, 2011; pp. 1–5.
22. Azmi, M.S.; Omar, K.; Nasrudin, M.F.; Muda, A.K.; Abdullah, A. Arabic calligraphy classification using triangle model for Digital Jawi Paleography analysis. In Proceedings of the 2011 11th International Conference on Hybrid Intelligent Systems (HIS), Malacca, Malaysia, 5–8 December 2011; pp. 704–708.
23. Adam, K.; Al-Maadeed, S.; Bouridane, A. Letter-based classification of Arabic scripts style in ancient Arabic manuscripts: Preliminary results. In Proceedings of the 2017 1st International Workshop on Arabic Script Analysis and Recognition (ASAR), Nancy, France, 3–5 April 2017; pp. 95–98. [CrossRef]
24. Allaf, S.R.; Al-Hmouz, R. Automatic Recognition of Artistic Arabic Calligraphy Types. *J. King Abdulaziz Univ.* **2016**, *27*, 3–17. [CrossRef]
25. Al-Hmouz, R. Deep learning autoencoder approach: Automatic recognition of artistic Arabic calligraphy types. *Kuwait J. Sci.* **2020**, *47*, 3.
26. Khayyat, M.; Elrefaei, L. A Deep Learning Based Prediction of Arabic Manuscripts Handwriting Style. *Int. Arab. J. Inf. Technol.* **2020**, *17*, 702–712. [CrossRef]
27. Kaoudja, Z.; Kherfi, M.L.; Khaldi, B. An efficient multiple-classifier system for Arabic calligraphy style recognition. In Proceedings of the 2019 International Conference on Networking and Advanced Systems (ICNAS), Annaba, Algeria, 26–27 June 2019; Institute of Electrical and Electronics Engineers (IEEE): Piscataway, NJ, USA, 2019; pp. 1–5.
28. Moghaddam, R.F.; Cheriet, M.; Milo, T.; Wisnovsky, R. A prototype system for handwritten sub-word recognition: Toward Arabic-manuscript transliteration. In Proceedings of the 2012 11th International Conference on Information Science, Signal Processing and their Applications (ISSPA), Montreal, QC, Canada, 2–5 July 2012; pp. 1198–1204.
29. Otsu, N. A threshold selection method from gray-level histograms. *IEEE Trans. Syst. Man Cybern.* **1979**, *9*, 62–66. [CrossRef]

30. Cheriet, M.; Kharma, N.; Liu, C.-L.; Suen, C. *Character Recognition Systems: A Guide for Students and Practitioners*; John Wiley & Sons: Hoboken, NJ, USA, 2007.
31. Lutf, M.; You, X.; Cheung, Y.-M.; Chen, C.P. Arabic font recognition based on diacritics features. *Pattern Recognit.* **2014**, *47*, 672–684. [CrossRef]
32. Hu, M.-K. Visual pattern recognition by moment invariants. *IEEE Trans. Inf. Theory* **1962**, *8*, 179–187. [CrossRef]
33. Khaldi, B.; Aiadi, O.; Kherfi, M.L. Combining colour and grey-level co-occurrence matrix features: A comparative study. *IET Image Process.* **2019**, *13*, 1401–1410. [CrossRef]