A Novel Lexicon Reduction Method For Arabic Handwriting Recognition

Safwan Wshah, Venu Govindaraju
Department of Computer Science and Engineering
University at Buffalo
Amherst, NY, USA
{srwshah, govind}@buffalo.edu

Yanfen Cheng¹ and Huiping Li²
¹Wuhan University of Technology, School of Computer Science, China
²Applied Media Analysis, Inc, USA
{chengyanfen,huipingli}@gmail.com

Abstract—In this paper, we present a method for lexicon size reduction which can be used as an important pre-processing for an off-line Arabic word recognition. The method involves extraction of the dot descriptors and PAWs (Piece of Arabic Word.) Then the number and position of dots and the number of the PAWs are used to eliminate unlikely candidates. The extraction of the dot descriptors is based on defined rules followed by a convolutional neural network for verification. The reduction algorithm makes use of the combination of two features with a dynamic matching scheme. On IFN/ENIT database of 26459 Arabic handwritten word images we achieved a reduction rate of 87% with accuracy above 93%.

Keywords- Lexicon reduction; arabic offline handwritten; handwritten recognition.

I. INTRODUCTION

All manuscripts must be in English. These guidelines include complete descriptions of the fonts, spacing, and related information for producing your proceedings.

The Arabic letters were created around the 7th century by adding dots to existing letters. Therefore several letters have exactly the same base form and differ only by single, double or triple dots [1]. Other small marks (diacritics) are used to indicate short vowels, but are often not used. Also the Arabic script, both handwritten and printed, is cursive and joined together except the sub-word ending with one of the ten red letters in Figure 1. 15 of 28 Arabic characters are joined together along a writing line [2]. The Arabic alphabet includes 28 letters (Figure 1), each with two or four shapes depending on the position it stays in a sub-word: start, middle, end of a sub-word, or alone. The Arabic text is written from right to left, and adjacent letters are joined together along the sub-word ending with one of the six red letters in Figure 1. 15 of 28 Arabic characters are dotted, with ten characters having one dot, three having two dots and two having three dots. These characters contain a unique main stroke, and are only distinguished by the presence/absence, position or number of dots.

![Figure 1. (a) The Arabic alphabet.](image)

Research in off-line handwritten word recognition has traditionally concentrated on relatively small lexicons from ten to a thousand words. The Arabic language has large lexicons containing 30,000 to 90,000 words [1]. Recognition with a large lexicon can be made more efficient by initially eliminating lexicon entries that are unlikely to match the given image. This process is called lexicon reduction or lexicon pruning, and can be used as an important pre-processing to improve the recognition accuracy and speed by removing classifier confusion [3].

A. Measuring lexicon reduction performance

Given a set of n word images and a corresponding lexicon, we denote the lexicon corresponding to image xi by Li. A lexicon reduction algorithm takes xi and Li as input and determines a reduced lexicon Qi ⊆Li. We denote the event that ti is contained in the reduced lexicon by a random variable A, where A=1, if ti∈Qi; and A=0, otherwise. The extent of reduction is captured by random variable R, defined as R=|Li|-|Qi|/|Li| [4].

Three measures of lexicon reduction performance are defined as follows;

- Accuracy of reduction: \( \alpha = E(A) \).
- Degree of reduction: \( \rho = E(R) \).
- Reduction efficacy: \( \eta = \frac{\rho}{k} \).

Note that \( \alpha, \rho, \eta \in [0,1] \). The accuracy and degree of reduction are usually related inversely to each other. \( \alpha \) can often be made arbitrarily close to unity at the expense of \( \rho \). The two measures are combined into one overall measure \( \eta \). The emphasis placed on accuracy relative to the degree of reduction is expressed as a constant k, which in turn may be determined by a particular application [4][7].

B. Background: Dot and PAW features

Some Arabic characters have exactly the same base forms and are distinguished only by presence, position and the number of dots. Double dots are often written as one
connected dot, which appears as a slash or hyphen [2]. Also, for letters with triple dots they can be written as a horizontal line with a single dot above, or an open curve to any of the four directions, as shown in Table 1. Table 1 shows the variations in handwritten dots [7].

<table>
<thead>
<tr>
<th></th>
<th>Printed</th>
<th>Handwritten</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Dot</td>
<td>·</td>
<td>·</td>
</tr>
<tr>
<td>Double Dot</td>
<td>..</td>
<td>ت</td>
</tr>
<tr>
<td>Triple Dot  .</td>
<td>.</td>
<td>٧ ٧ ٧ ٧ ٧ ٧</td>
</tr>
</tbody>
</table>

Table 1: Variations of handwritten dots.

Besides the dots, an Arabic word consists of one or more PAWs (or Sub-words). A word with three PAWs can be easily differentiated from another word with only one PAW, as shown in Figure 2. These features can be used to reduce the lexicon size [7].

<table>
<thead>
<tr>
<th>شعار السراي الهلالي</th>
<th>(a)</th>
<th>(b)</th>
<th>(c)</th>
</tr>
</thead>
</table>

Figure 2. Word of different number of PAWs (a) one PAW, (b) four PAWs, and (c) two PAWs.

Arabic lexicon reduction has been used by other researchers as well. Mozaffari [7] [8] described an approach which is similar to ours in terms of the use of PAWs and dot descriptors. However, they had difficulty in detecting text baseline which lead to improper dot descriptor positions.

II. OUR APPROACH

As described above, our lexicon reduction algorithm is based on two features: the number of PAWs and the dot descriptors. After extracting these features we then use a dynamic matching approach to perform lexicon reduction.

A. Extracting Dot Descriptors

Three steps are used to extract the dot descriptors as shown in figure 3. We start from a rule-based step to find all candidate segments that could be dot descriptors. We then use a convolutional neural network (CNN) to verify these dot candidates. The last step checks the positions of dot descriptors geometrically and combines all dots together to form the final string of top and bottom dot descriptors.

![Dot descriptor recognition stages](image)

Figure 3. Dot descriptor recognition stages, with red color representing all dot candidates detected by the rule-based stage, the blue color representing the non-dot descriptor eliminated by CNN or geometrical based rules, and the green box representing the result of combined dots.

We first extract the connected components and assign the small intersected segments as dot candidates. The rules are based on the size of the connected component and the percentage of intersection. Figure 4 shows some examples of the dot segments detected by this stage.

![Initial detection](image)

Figure 4. The initial dot detection (in red color).

2) Neural Network based on refining

Some isolated small characters and other shapes will be detected as dots as shown in Table 2 and thus a robust classifier is needed to refine the output from the first stage. A convolutional neural network was built to refine the shapes detected by the rule-based stage.

<table>
<thead>
<tr>
<th></th>
<th>Printed</th>
<th>Handwritten</th>
</tr>
</thead>
<tbody>
<tr>
<td>٧ ٧ ٧ ٧ ٧ ٧</td>
<td>٧ ٧ ٧ ٧ ٧ ٧</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: non-dot segments detected by the rule based stage.

A convolutional neural network (CNN) is a feed-forward neural network tailored for minimizing the sensitivity to translation [5], local rotation and distortion of the input image with three factors: local receptive fields, shared weights, and spatial sub-sampling. The use of shared weights also reduces the number of parameters in the system which aids in generalization and accelerates the speed.

CNNs have been successfully applied to handwritten character recognition with very good results reported [6].

2866
A convolutional neural network of eleven classes which represent the majority of the shapes detected by rule based on dot descriptor was trained as shown in figure 5, where 300 sample images are used for training for each class.

Figure 5. Convolutional Neural network for refining rule based detected objects.

3) Combination and Geometrical based stage

After CNN based refining most of the non-dots are removed. The last stage combines the dots that have been written as a group of separated dots based on geometrical rules between the detected descriptors. One example of these defined geometrical rules is that a triple dot never appears at the bottom of a word. Figure 3 shows an example of the combination stage where the green boxes combine the separated dots and use geometrical rules to remove the chadda above the two dots in the middle of the word.

The position of a dot represents important information for lexicon reduction. If the vertical strip pixels of a dot descriptor are above the non-dot segment it is considered as a top dot descriptor; otherwise it was considered as a bottom dot as shown if figure 6.

Figure 6. Top and bottom dot descriptors, with red representing top dot and blue representing lower dot. Green color represents the intersected vertical strip area.

B. PAW Detection

After extracting the dot descriptor features, we detect the number of PAWS in an Arabic word as the second category feature. The number of PAWs can be easily counted after the detection of dots and non-characters shapes as addressed in previous stages. The PAWs are detected by counting all connected objects after removing all dots and non character shapes detected at the first stage.

C. Lexicon Reduction

Lexicon is reduced by combining the PAW counts and the dot descriptors. The subset LR represents the reduced lexicon containing the most similar entries in lexicon L to the input image [7]. Assume the estimated number of PAWs for a given word is P. Then only word entries with P−t[P] to P+t[P] sub-words are taken into account and all other words in the lexicon are eliminated, where t is the distance cost (difference) table between the number of the detected PAWs of the input image and the PAWs of word entries in the lexicon. The table cost value is dynamically defined where smaller PAW count has lower cost value and higher PAW count has larger cost value.

We then use the dot information to interpret the input image as a string of top and bottom dot descriptors and match with the descriptors of all lexicon entries. The results of the lowest cost difference with specified threshold are considered as the reduced lexicon. In this paper, Table 3 shows the cost of each edit operation based on experiments performed on different threshold values.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>.</td>
<td>~</td>
<td>v,&gt;,^</td>
</tr>
<tr>
<td>.</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>~</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>v,&gt;,^</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 3: edit operation cost between dot descriptors.

Note that the madda (~) edit operations have low cost values because it appears as double dot, triple dot or non-dot descriptor sometimes. The cost values in table 3 increases by 2 if the two edit operation descriptors are in different positions, i.e. top and bottom positions.

Although either PAW or dot descriptors can be used separately to reduce the lexicon, the combination of these two features achieve the better result. In experiments we found the best result was achieved when PAW based lexicon reduction was applied first followed by dot descriptor matching.
III. Experimental Results

In our experiments, we used IFN/ENIT database that contains 26459 Arabic handwritten word images of 937 Tunisian town and village names. Figure 7 shows the lexicon reduction results of PAW, dot descriptor matching, and the combination of the two methods respectively. The experiments were performed on different values of pruning thresholds. The combination of both methods shows the best result which achieves up to 93% accuracy with 87% reduction.

The lexicon reduction based on the dot descriptor matching (red color in Figure 7) reduces the lexicon two times more than the PAW based method (blue color in Figure 7) with the same accuracy. And it achieves 95% accuracy with 67% reduction. This is due to more features in dot descriptors which include shapes and positions. For different values of pruning threshold the average reduction rate based on dot descriptor matching is 64.2% with corresponding 96.2% accuracy.

On the other hand the PAW based reduction reduces the lexicon up to 40% with a little bit higher accuracy of 97%. The average reduction rate for the PAW based method is 35.7% with corresponding 97.5% accuracy.

The combination of the two methods provides better results due to the high accuracy from the PAW and the high reduction rate from the dot descriptor matching. The combination achieves an average reduction rate of 85.6% with corresponding 94.6% accuracy, and 93.4% accuracy in average with corresponding 85.3% reduction rate.

On IFTN dataset Mozaffari[7] showed 92.5% reduction with 74% accuracy. We achieved higher accuracy for the same reduction ratio (88% accuracy) due to the robustness of the conventional neural network, dynamic matching, and making use of both dot descriptors and the PAW counts after dot recognition.  

IV. Conclusion and Future Work

Lexicon reduction is used to improve recognition accuracy and accelerate the process by trimming down unnecessary entries in the lexicon. In this paper we presented a technique to reduce the lexicon based on two unique features in Arabic language: PAW and dot descriptors. Both methods have been used separately, and then combined to improve the accuracy with higher reduction rate. The proposed technique is efficient due to the simple applied rules and pruned weights of the convolutional neural network. In the future work we will study the effect of applying lexicon reduction methods presented in this paper to a full Arabic handwriting recognition system in term of accuracy and speed. We will report the results in our following research.

REFERENCES