International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:3, No:6, 2009

Comparing Arabic and Latin Handwritten Digits Recognition Problems

Sherif Abdelazeem
shazeem@aucegypt.edu
Electronics Engineering Dept., The American University in Cairo

Abstract—A comparison between the performance of Latin and Arabic handwritten digits recognition problems is presented. The performance of ten different classifiers is tested on two similar Arabic and Latin handwritten digits databases. The analysis shows that Arabic handwritten digits recognition problem is easier than that of Latin digits. This is because the interclass difference in case of Latin digits is smaller than in Arabic digits and variances in writing Latin digits are larger. Consequently, weaker yet fast classifiers are expected to play more prominent role in Arabic handwritten digits recognition.

Keywords—Handwritten recognition, Arabic recognition, Digits recognition, Document recognition

I. Introduction

Handwritten digit recognition problem can be seen as a subtask of the more general Optical Character Recognition (OCR) problem. However, there are some applications (e.g., postal code and bank checks reading) that are restricted to recognizing digits but require very high accuracy and speed. While recognition of handwritten Latin digits has been extensively investigated using various techniques [1], too little work has been done on Arabic digits.

First, the issue of the names "Arabic digits" and "Latin digits" has to be resolved because naming conventions for different numeral systems may be confusing. Digits used in Europe and several other countries are sometimes called "Arabic Numbers"; and digits used in the Arab world are sometimes called "Hindi Numbers". A different naming convention is used in this paper. Digits used in Europe will be referred to as "Latin Digits" and those used in the Arab world will be referred to as "Arabic Digits". It is worthwhile mentioning here that Arabic and Persian handwritten digits (digits used in Iran) are similar but not identical. Table I shows Arabic handwritten digits with different writing styles as well as their printed versions.

This paper compares the performances of different classification techniques on both Arabic and Latin handwritten digit recognition problems. It is well known that there is no universally powerful or universally weak classifier [2]. The performance of a certain classification technique depends on how it matches the problem at hand. Realizing this, comprehensive comparative studies [3] compare the performances of classifiers on many different problems. However, if researchers doing such comparative studies do not have direct experience with the problems they use, they may not be able to give reasons. why a certain classification technique performs poorly on a certain problem while having a good performance on another. This paper attempts to understand the difference in performance of various classifiers when applied to the two different recognition problems of "Arabic" versus "Latin" handwritten digits. This should give researchers the chance to gain more insight into the natures of the two problems and, thus, be better able to do classifier-problem matching.

TABLE I
Arabic Printed and Handwriten Digits.

Latin Equivalent	0	1	2	3	4	5	6	7	8	9
Printed	٠	١	٢	٣	٤	٥	٦	٧	٨	٩
Typical Handwritten	٠	١	٢	٣	٤	٥	٦	٧	٨	٩
Other Writing Style	–	–	–	٣	–	–	–	–	–	–

The remaining of the paper is organized as follows. Section II is a survey of previous research on Arabic handwritten digits recognition problem that motivated this study. The classifiers and databases used in the comparison are described in Section III. Section IV reports and analyzes the performances of the classification techniques on both the Latin digits and Arabic digits. Finally, the conclusion is in Section V.

II. Motivation

Little previous work has been done on Arabic handwritten digits recognition. Al-Omari et al. [4] proposed a system for recognizing Arabic digits from '1' to '9'. They used a scale-, translation-, rotation-invariant feature vector to train a probabilistic neural network (PNN). Their database was composed of 720 digits for training and 480 digits for testing written by 120 persons. They achieved 99.75% accuracy. Said et al. [5] used pixel values of the 16x20 size-normalized digit images as features. They fed those values to an Artificial Neural Network (ANN) with a dynamic number of hidden units. They used a training set of 2400 digits and a testing set of 200 digits written by 20 persons to achieve 94% accuracy. It has to be noted that the results of different works can not be compared because the used databases do not have the same sizes or formats. Abdelazeem et al. [6] did a comprehensive study on the problem of Arabic handwritten digit recognition problem. They introduced a large binary Arabic handwritten Digits dataBase(the ADBase) and a modified gray level version of it (Modified ADBase or MADBase) and studied the performance of various classifiers/features combinations on the Arabic handwritten digit recognition problem. The performances of well known feature extraction techniques followed by various classifiers have been analyzed

in an approach similar to that followed by Liu et al. [1] in their study of Latin handwritten digit recognition problem. Both ADBase and MADBase are available online for free at *http://datacenter.aucegypt.edu/shazeem/*.

Comparing the results obtained in [6] for Arabic handwritten digits recognition problem with those in [1] for the Latin handwritten digit recognition problem reveals an interesting remark; and that is while the performance of a powerful classifier such as SVM is strong on both problems, the performance of less powerful classifiers such as linear or neural classifiers is better on the Arabic problem than on the Latin one. It is this observation that motivated the present comparative study in an attempt to understand the similarities/differences between the two problems.

III. The classification comparison

The comparison between the Arabic and the Latin handwritten digit recognition problems is done by studying the performances of different classification techniques on Arabic as well as Latin handwritten digits on the raw pixel values of the digit images without any feature extraction. To make such a comparison valid, the used databases for Arabic and Latin digits should be of the same format. MNIST [7] is the used Latin database while MADBase [6] is used as the Arabic database because it has the same size and format as MNIST [7]. The present study uses 10 different classification techniques to compare between the MNIST and MADBase databases: LeNet 5 [7], Prazen window using RBF kernel [2], artificial neural network (ANN), two different k-nearest neighborhood classifiers using k = 3 and k = 5, principle component analysis (PCA) with quadratic classifier, PCA with neural network classifier [2], One versus All (OVA) linear classifiers, One versus One (OVO) linear classifiers, OVO SVM using linear kernel and finally OVO SVM using RPF kernel. Each of those 10 classifiers is fed directly with gray-scale pixel values of digit images without feature extraction step. Some of the classifiers have parameters which need to be specified (e.g., the number of hidden units of ANN). Such parameters are selected so as to give the best performance on a validation set; which is a set of 10,000 digits selected randomly from the training set. The performances of many of the classifiers used in this study have been evaluated on MNIST previously in [7]. However, their performances are re-evaluated in this study using the same classifiers' implementations used with the MADBase to ensure the validity of the comparison. Classification techniques used in this study are implemented in C++ using Torch machine learning library.

IV. Latin digits versus Arabic digits

Table II reports the results of different classifiers on both Arabic and Latin digits. The performances of different classification techniques are compared using 3 metrics. *The Accuracy (ACC)* which is the percentage of test samples that are classified correctly, the *Rejection Rate (RR)* which is the percentage of test samples that should be rejected to reach an accuracy of 99.5%, and the *timing performance (T)* which is measured by the number of CPU seconds needed to recognize the 10,000 samples of the test set relative to the fastest classifier. Fig.1 displays a bar graph for the results for better visualization of the accuracy comparison.

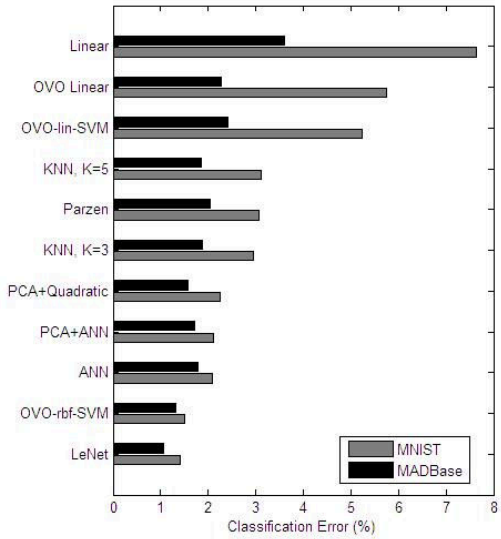


Fig. 1. Comparison of 10 classification techniques on MNIST and MADBase

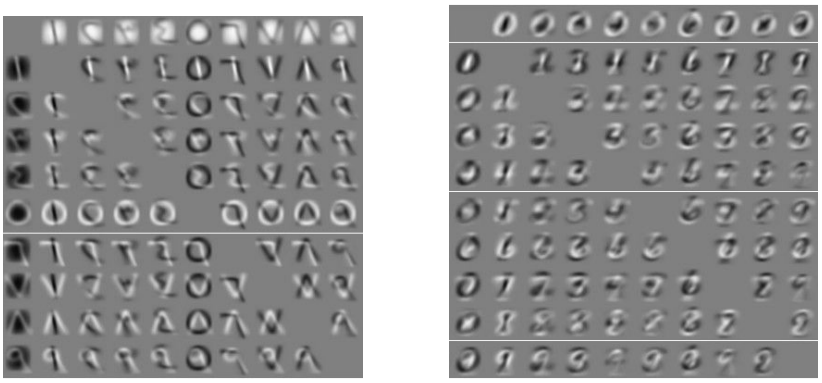
Note that the order of arranging the classifiers in Fig.1 is not arbitrary. They are arranged in a descending order form top to bottom according to their classification errors on MNIST. It is clear from Fig.1 that classification errors of different classifiers on MADBase follow the same pattern of decay as the case of MNIST. The most interesting observation that may be noticed from Fig.1 is that although classification errors of the most powerful classifiers in case of MNIST compared with MADBase, there is a huge gap between classification errors of the weakest classifiers in case of MNIST and their classification error for MADBase.

An explanation for this may be as follows. The Arabic digits recognition problem is in general easier than that of Latin digits. This is why simple classifiers (like linear and OVO linear) score high on MADBase. However, the percentage of the poorly written digits are close for MADBase and MNIST (both around 1%) and they are equally hard to classify for the weak and the powerful classifiers. See Fig.2(a) for the patterns misclassified by LeNet for Arabic digits and Fig.2(b) for Latin digits. But why is the Arabic digits recognition problem simpler than that of Latin digits? Two reasons can explain this. The first one is that the interclass distance between each pair of typical (i.e., not poorly written) Latin digits tend to be smaller than that between pairs of Arabic digits. When interclass distance between two digits is small, any variation in writing one digit will lead this digit to be easily confused with the other.

The second reason why Latin digits are harder to classify than Arabic digits is that the variance in writing each of Latin digits is in general higher than that in case of Arabic. In the following two subsections, each of these two proposed reasons is discussed in details.

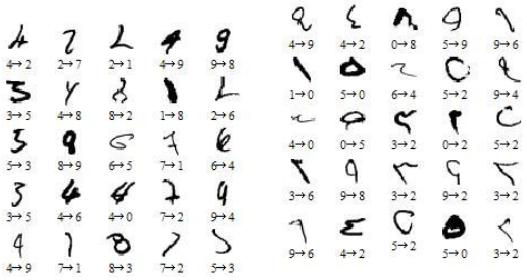
TABLE II
PERFORMANCES OF DIFFERENT CLASSIFICATION TECHNIQUES ON MNIST AND MADBASE

Classifiers	Latin			Arabic		
	Acc(%)	RR(%)	T	Acc(%)	RR(%)	T
OVA Linear	92.35	41.34	1	96.4	13.49	1
OVO Linear	94.25	43.83	5	97.72	10.38	5
OVO-lin-SVM	94.76	27.61	5	97.59	8.48	5
KNN, K=3	97.05	15.33	7100	98.13	18.38	7100
KNN, K=5	96.88	12.47	7100	98.14	4.9	7100
Parzen	96.93	12.4	8150	97.96	5.44	8150
PCA+Quadratic	97.75	7.72	8	98.42	3.31	8
PCA+ANN	97.88	5.27	10	98.28	4.28	10
ANN	97.91	4.8	46	98.21	5.33	16
OVO-rbf-SVM	98.5	2.93	2120	98.78	2.34	1163
LeNet	98.9	1.76	90	98.91	1.75	90



(a) Arabic Digits (b) Latin Digits

Fig. 3. Image at row i and column j is the result of subtraction digit j from digit i for (a) Arabic digits, and for(b) Latin digits.



(a) Latin Digits (b) Arabic Digits

Fig. 2. Samples of hard-to-classify for (a) Latin digits, and for(b) Arabic digits.(misclassified by LeNet)

A. Interclass distances

It is noticed that Latin digits have so many common strokes, and, for many cases, any addition or deletion of a stroke from a Latin digit may lead it to appear as another one. For example, adding a stroke to the upper right of Latin digit '5' will lead it to appear like '9'; and to appear like '6', if a stroke is added at the left bottom of it. Also adding a stroke at the left bottom of '9' leads it to appear like '8'. Digit '4' already appears like '9', and removing the upper stroke of '9' leads it to appear like another writing style of '4'. The fact that Latin digits have many common strokes helps Latin digits to be represented using only 7 strokes in a 7- segment display with natural appearance. But looking at Arabic digits in Table I, it is noted that there are not so many common strokes between them, and any attempt to represent them using 7-segment will lead to very artificial appearance (a dot matrix representation is usually used for them in practice).

A visual illustration of this idea is depicted in Fig. 3 which shows the resulting images from subtracting the mean of each digit from the means of other digits for both Latin and Arabic digits. The image at row i and column j is the result of subtracting the mean of digit j from that of digit i; where foreground (white) is represented by '1' and background (black) is represented by '0'. Each pixel resulting from this subtraction is then added to 1, and then the result is divided by 2. Hence, if the result of subtraction for some pixel location is '-1', it appears as black; if '0', it appears as gray; and if '1', it appears as white. Now, if any two digits have a common stroke, the average of their subtraction will be '0' at the location of this stroke; and hence, will appear gray. Non-common strokes appear black or white depending on which digit is subtracted from the other. It is clear from Fig.3 that Latin digits have many common strokes and hence the gray pixels dominate to the degree that most of the digit pairs are not distinguishable from each other. On the other hand, for

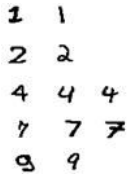


Fig. 4. Some Latin digits and their different typical writing styles.

Arabic digits, most of digit pairs are quite distinguishable. The above is rather an informal evidence. In the following,a more formal numerical evidence is going to be introduced. A numerical measure for the interclass distance between a pair of classes (i, j), i.e. between digit i and digit j, may be the Euclidean distance d_{ij} between their means,

$$d_{ij} = \sqrt{\sum_{\forall x}\sum_{\forall y}[\mu_i(x,y) - \mu_j(x,y]^2}$$

where, $i, j = 0, 1, , 9$; $x, y = 1, 2, , 28$; and $\mu_i(x, y)$ is the mean value of the pixel at location (x, y) of the images of digit k,

$$\mu_k(x,y) = \frac{1}{N}\sum_{n=1}^{N_k} I_{kn}(x,y)$$

where $I_{kn}(x, y)$ is the pixel value at location (x, y) of the n^{th} image of the training set of digit k, and N is the number of training samples of digit k. A numerical measure for the overall separation between all class pairs ($d_{overall}$) may be the summation of d_{ij} for all i and j, i.e.,

$$\mathrm{d_{overall}} = \sum_{\forall i}\sum_{\forall j} d_{ij}$$

The value of $d_{overall}$ was found for Arabic digits to be larger than that of Latin by 11.9%. This means that interclass separations between Arabic digits are larger than that in case of Latin. To check the significance of this result, the hypothesis of having the $d_{overall}$ equal for both cases Latin and Arabic is checked. To do so, the training set is partitioned into 6 separate subsets, each containing 10,000 samples. The $d_{overall}$ measurement is calculated for each subset resulting in 6 different measurements. This is done for both cases Latin and Arabic. A t-test [8] is then performed to check the hypothesis that Latin and Arabic have equal mean value of $d_{overall}$. The hypothesis was found to have a p-value of 4.3×10^{-8}, which means that it can be rejected with very small significance level.

B. Variances

The second reason why Latin digit recognition problem is harder than that of Arabic is that variances in writing Latin digits are larger than that of Arabic. One measure of variances (scattering) S_k of some pattern k around its mean is the trace of its covariance matrix [2],

$$S_k = tr\left[\sum\nolimits_k\right] = \sum_{i=1}^{d}\sigma_{ki}^2,$$

,where $\sum_k$ is the covariance matrix of pattern k, σ_{ki}^2 is the variance of the i^{th} dimension of pattern k, and d is the dimensionality of the pattern. For digit recognition problem, the scattering of some digit k around its mean is then given by,

$$S_k = \frac{1}{N_k}\sum_{n-1}^{N_k}\sum_{\forall x}\sum_{\forall y}[I_{kn}(x,y) - \mu_k(x,y)]^2$$

A numerical measure for the overall scattering is $S_{overall} = \sum_{k=0}^{9} S_k$. It is found that $S_{overall}$ for Latin digits is 27.9% larger than that of Arabic. This means that large deviation from the mean is more probable for Latin digits than that for Arabic. The reason behind this might be the fact that Latin digits have many writing styles as shown in Fig. 4.

To check the significance of the $S_{overall}$ comparison between Arabic and Latin digits, the training set is partitioned into 6 separate subsets, each containing 10,000 samples. The $S_{overall}$ measurement is calculated for each subset resulting in 6 different measurements. This is done for both cases Latin and Arabic. A t-test [8] is then performed to check the hypothesis that Latin and Arabic have equal mean value of $S_{overall}$. The hypothesis was found to have a p-value of 1.1×10^{-11}. This means that the hypothesis can be rejected with very small significance level.

V. Conclusion

A comparison between the performances of 10 classifiers is made on both Arabic and Latin handwritten digit recognition problems. An interesting result of such comparison is that while the performances of the most powerful classifiers on Arabic and on Latin digits are close, the weaker classifiers show very good performance on Arabic digits unlike their performance on Latin digits. The reason behind this is suggested to be that Arabic handwritten digits recognition problem is in general easier than that of its Latin counterpart. An analysis is given for the reasons why Arabic digits are in general easier to classify than Latin digits and it has been found that this is because interclass distances between Arabic digits are larger than those of Latin and variances in writing Arabic digits are smaller.

The practical implications of the findings of the comparison presented in this paper become evident in the case of the Arabic handwritten digits recognition problem in applications where classification speed is a concern. In those applications, weaker classifiers may present a plausible alternative to powerful ones. In classification cascades for example, weaker yet faster classifiers play prominent role at the early stages of the cascade to speed up the performance of the whole cascade [9], [10] It is expected that weaker classifiers play a more significant role in Arabic handwritten digits classification cascades than their role in the Latin case due to their superior performance in Arabic digits classification.

References

[1] Cheng-Lin Liu, Kazuki Nakashima, Hiroshi Sako, and Hiromichi Fujisawa. Handwritten digit recognition: benchmarking of state-of-the-art techniques. *Pattern Recognition*, 36(10):2271– 2285, Oct 2003.

[2] D. Stork R. Duda, P. Hart. *Pattern Classification*. Wiley, New York, 2nd edition edition, 2000.

[3] Rich Caruana and Alexandru Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *International Conference on Machine Learning (ICML)*, pages 161–168. Department of Computer Science, Cornell University, 2006.
[4] Faruq A. Al-Omari and Omar M. Al-Jarrah. Handwritten indian numerals recognition system using probabilistic neural networks. *Advanced Engineering Informatics*, 18(1):9–16, 2004.
[5] Fady N. Said, Rita A. Yacoub, and Ching Y. Suen. Recognition of english and arabic numerals using a dynamic number of hidden neurons. In *ICDAR*, pages 237–240, 1999.
[6] Sherif Abdelazeem and Ezzat El-Sherif. Arabic handwritten digit recognition. *International Journal of Document Analysis and Recognition IJDAR*, 11(3):127–141, Dec 2008.
[7] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. volume 86, pages 2278–2324. IEEE, nov 1998.
[8] E. Kreyszig. *Advanced Engineering Mathematics*. Wiley, New York, 9th edition edition, 2006.
[9] Ezzat El-Sherif, Sherif Abdelazeem, and M. Yazeed. Automatic generation of optimum classification cascades. In *19th International Conference on Pattern Recognition (ICPR 2008)*, Tampa, FL, Dec 2008.
[10] Sherif Abdelazeem. A greedy approach for building classification cascades. In *The Seventh International Conference on Machine Learning and Applications (ICMLA'08)*, pages 115–120, San Diego, CA, Dec 2008.