Structural Description to Recognising Arabic Characters Using Decision Tree Learning Techniques

Adnan Amin

School of Computer Science, University of New South Wales
Sydney, 2052, Australia
amin@cse.unsw.edu.au

Abstract: Character recognition systems can contribute tremendously to the advancement of the automation process and can improve the interaction between man and machine in many applications, including office automation, cheque verification and a large variety of banking, business and data entry applications. The main theme of this paper is the automatic recognition of hand-printed Arabic characters using machine learning. Conventional methods have relied on hand-constructed dictionaries which are tedious to construct and difficult to make tolerant to variation in writing styles. The advantages of machine learning are that it can generalize over the large degree of variation between writing styles and recognition rules can be constructed by example.The system was tested on a sample of handwritten characters from several individuals whose writing ranged from acceptable to poor in quality and the correct average recognition rate obtained using cross-validation was 87.23%.

Keywords: Pattern Recognition, Arabic characters, Hand-printed characters, Parallel thinning, Feature extraction, Structural classification, Machine Learning C4.5

1 Introduction

Character recognition is commonly known as Optical Character Recognition (OCR) which deals with the recognition of optical characters. The origin of character recognition can be found as early as 1870 [1] while it became a reality in the 1950's when the age of computer arrived [2]. Commercial OCR machines and packages have been available since the mid 1950's. OCR has wide applications in modern society: document reading and sorting, postal address reading, bank cheque recognition, form recognition, signature verification, digital bar code reading, map interpretation, engineering drawing recognition, and various other industrial and commercial applications.

Much more difficult, and hence more interesting to researchers, is the ability to automatically recognize handwritten characters [3]. The complexity of the problem is greatly increased by noise and by the wide variability of handwriting as a result of the mood of the writer and the nature of the writing. Analysis of cursive scripts requires

T. Caelli et al. (Eds.): SSPR&SPR 2002, LNCS 2396, pp. 152-158, 2002.

the segmentation of characters within the word and the detection of individual features. This is not a problem unique to computers; even human beings, who possess the most efficient optical reading device (eyes), have difficulty in recognizing some cursive scripts and have an error rate of about 4% on reading tasks in the absence of context [4].

Different approaches covered under the general term 'character recognition' fall into either the on-line or the off-line category, each having its own hardware and recognition algorithms.

Many papers have been concerned with Latin, Chinese and Japanese characters, However, although almost a third of a billion people worldwide, in several different languages, use Arabic characters for writing, little research progress, in both on-line and off-line has been achieved towards the automatic recognition of Arabic characters. This is a result of the lack of adequate support in terms of funding, and other utilities such as Arabic text database, dictionaries, etc..[5]

This paper proposes a structural method for the extraction of line and curve primitive features. Such features will then be represented in attribute/value form, which is then input to the inductive learning system, C4.5 [6], to generate a decision tree. This decision tree can then be used to predict the class of an unseen character.

Fig. 1 depicts a block diagram of the system.

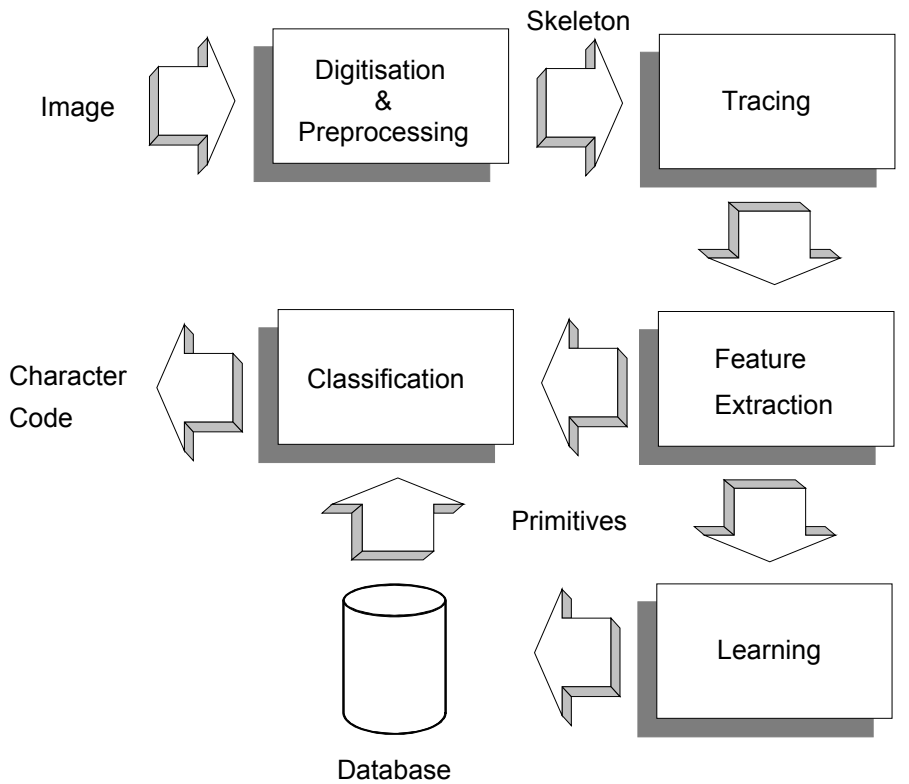

Fig. 1. Block diagram of the system

2 Feature Extraction

The characters are digitized using a 300 dpi scanner, pre-processed and thinned using one pass parallel thinning algorithm [7] The simple structural information such as lines, curves and loops that describe the characters are extracted by tracing the thinned image with some pre-defined primitives as shown in Fig. 2. A detailed description of the feature extraction process is given in [8]. The characters are then classified using a primary classifier and an exception classifier. The primary classifier uses a machine learning program (C 4.5) which uses an induction algorithm for the generation of the classification rules.

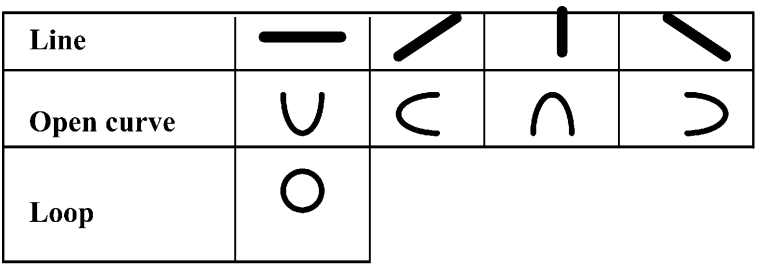

Fig. 2. Primitive features used in this system

3 Classification Using C4.5

C4.5 [6, 9], is an efficient learning algorithm that creates decision trees to represent classification rules. The data input to C4.5 form a set of examples, each labeled according to the class to which it belongs. The description of the example is a list of attribute/value pairs. A node is a decision tree represents a test on a particular attribute. Suppose its color, size and shape describe an object where color may have values red, green or blue; size may be large or small and shape may be circle or square. If the root node of the tree is labeled color then it may have three branches for each color value. Thus if we wish to test an objects color and it is red, the object descends the red branch. Leaf nodes are labeled with class names. This when an object reaches a leaf node, it is classified according to the name of the leaf node.

Building a decision tree proceeds as follows. The set of all examples forms an initial population. An attribute is chosen to split the population according to the attribute's values. Thus, if color is chosen then all red objects descend the red branch, all green objects descend the green branch, etc. Now the population has been divided into sub-populations by color. For each sub-population, another attribute value is chosen to split the sub-population. This continues as long as each population contains a mix of examples belonging to different classes. Once a uniform population has been obtained, a leaf node is created and labeled with the name of the class of the population.

The key to the success of a decision tree learning algorithm depends on the criterion used to select the attribute to use for splitting. If attribute is a strong indicator of an example's class value, it should appear as early in the tree as possible. Most decision tree learning algorithms use a heuristic for estimating the best attribute. In

C4.5, Quinlan uses a modified version of the entropy measure from information theory. For our purposes, it is sufficient to say that this measure yields a number between 0 and 1 where 0 indicates a uniform population and 1 indicates a population where there is equal likelihood of all classes being present. The splitting criterion seeks to minimize the entropy.

A further refinement is required to handle noisy data. Real data sets often contain examples that are misclassified or which have incorrect attribute values. Suppose decision tree building has constructed a node which contains 99 examples from class 1 and only one example from class 2. According to the algorithm presented above, a further split would be required to separate the 99 from the one. However, the one exception may be misclassified, causing an unnecessary split. Decision tree learning algorithms have a variety of methods for "pruning" unwanted subtrees. C4.5 grows a complete tree, including nodes created as a result of noise. Following initial tree building, the program proceeds to select suspect subtrees and prunes then, testing the new tree on a data set, which is separate from the initial training data. Pruning continues as long as the pruned trees yield more accurate classifications on the test data.

The C4.5 system requires two input files, the *names* and the *data* files. The *name*s file contains the names of all the attributes used to describe the training examples and their allowed values. This file also contains the names of the possible classes. The classes are the Arabic words.

The C4.5 data files contains the attributes values for example objects in the format specified by the name file, where each example is completed by including the class to which it belongs.

Every Arabic character can be composed of at most five segments. A segment can be a complementary character, line, curve or loop. A 12-bit segment coding scheme is used to encode these segments. The scheme is depicted in Fig. 3.

The interpretation of the last seven bits of the code depends on the first five bits. However, only one of the first five should be 1. If zero or more than one bit is 1, the system automatically rejects the segment. For a segment to be misidentified, two bits would need to be incorrectly transmitted. Including the one that identifies the segment. Therefore, the encoding is relatively robust.

4 Experimental Results Using Cross-Validation

The first decision required in setting up a machine learning experiment is how the accuracy of the decision tree will be measured. The most reliable procedure is *cross-validation.* N-fold cross-validation refers to a testing strategy where the training data are randomly divided into *N* subsets. One of the *N* subsets is withheld as a *test* set and the decision tree is trained on the remaining *N-1,* subsets. After the decision tree has been built, its accuracy is measured by attempting to classify the examples in the test set. Accuracy simply refers to the percentage of examples whose class is correctly predicted by the decision tree. The error rate is the accuracy subtracted from 100%. To compensate for sampling bias, the whole process is repeated *N* times, where, in each iteration, a new test set is withheld and the overall accuracy is determined by averaging the accuracies found at each iteration.

Fu [10] describes the cross-validation process as "K-fold cross-validation (Stone[11]) repeats K times for a sample set randomly divided into disjoint subset, each time leaving one out for testing and others for training" The value of K = 10 is usually recommended [12]. Cross-validation requires that the original data set id split in K disjoint sets. At any one time, 90% of the data is used for training and the system is tested on the remaining 10%. At the end of 10 folds, all data has been tested. In every fold therefore the training and test patterns remain different. In brief, the cross-validation procedure for our purposes involves the following:

begin

1. For a total of 120 classes and 6000 patterns, interleave the patterns so that the pattern of class *j* is followed by a pattern of class *j*+1.
2. Segment data into K sets (k_1, k_2, ... k_K) of equal size, e.g. in our case for K = 10, we have 600 pattern in each set..
3. Train the C.4.5 with K-1 sets and test the system on the remaining one data set. Repeat this cycle K times, each time with a training set which is distinct from the test set.

   For i = 1 to K **do**
   Begin
   Train with data from all partitions k_n where $1 \leq n \leq K$, and $i \neq n$.
   Test with data from partition k_i.
   End;

4. Determine the recognition performance each time and take the average over a total of K performances.

end.

Table 1 shows the error rates performance using ten-fold cross-validation. It is important to note here that the system performs extremely well with recognition rates ranging between 85 % and 90% on different folds and the overall recognition is 87%. This is a very good performance taking into account the fact that we have a limited number of samples in each class.

Table 1. Error rates performance using ten fold cross validation

Fold	Error-Rate % Testing
1	12.35
2	13.89
3	14.23
4	10.14
5	11.56
6	12.90
7	11.67
8	13.76
9	14.20
10	11.45
Average	12.67

5 Conclusion

This paper presents a new technique for recognizing printed Arabic text and as indicated by the experiments performed, the algorithm resulted in a 87.23% recognition rate using the C4.5 machine learning system.

Moreover, the system used a structural approach for feature extraction (based on structure primitives such as curves, straight lines and loops in similar manner to which human begins describe characters geometrically). This approach is more efficient for feature extraction and recognition.

The study also shows that machine learning algorithms such as C4.5 are capable of learning the necessary features needed to recognize printed Arabic text achieving a best average recognition rate of 87.23% using ten fold cross-validation. The use of machine learning has removed the tedious task of manually forming rule-based dictionaries for classification of unseen characters and replaced it with an automated process which can cope with the high degree of variability which exists in printed and handwritten characters. This is very attractive feature and, therefore, further exploration of this application of machine learning is well worthwhile.

In the area of recognition, a structural approach has been previously used. This approach is sufficient to deal with ambiguity without using contextual information. This area remains undeveloped due to the immaturity of vital computational principles for Arabic character recognition.

References

1. V. Govindan and A. Shivaprasad, Character recognition- a review, Pattern Recognition, 23(7), pp. 671-683, 1990.
2. S. Mori, C. Y. Suen and K. Yamamoto, Historical review of OCR research and development, *Proceedings of the IEEE 80* (7), pp. 1029-1058, 1992.
3. E. Lecolinet E. and O. Baret, Cursive word recognition: Methods and strategies, Fundamentals in Handwriting Recognition, S. Impedovo, Ed. Springer-Verlag, 1994, pp. 235–263.
4. C. Y. Suen, R. Shingal and C. C. Kwan, Dispersion factor: A quantitative measurement of the quality of handprinted characters, Int. Conference of Cybernetics and Society, 1977, p.681–685.
5. A. Amin, Off_line Arabic characters Recognition: The State Of the Art, *Pattern Recognition 31(5),* 517-530, 1998.
6. Quilan J. R., C4.5 : programs for machine learning, San Mateo CA, Morgan Kauffman, 1993.
7. B.K. Jang and R.T. Chin, One-pass parallel thinning: analysis, properties, and quantitative evaluation, *IEEE Trans. Pattern Anal. Mach. Intell.* PAMI-14, pp. 1129-1140, 1992.
8. A. Amin, H. Al-Sadoun and S. Fischer, Hand-Printed Arabic Characters Recognition System Using an Artifial Network, *Pattern Recognition 29(4)*, pp. 663-675, 1996.
9. J. R. Quilan, Discovering rules for a large collection of examples, Edinburgh University Press, 1979.

10. L. Fu, Neural Networks in Computer Intelligence, McGraw-Hill, Singapore, pp. 331-348, 1994.
11. M. Stone, Cross-validatory choice and assessment of statistical predictions, *Journal of the Royal Statistical Society 36(1),* pp. 111-147, 1974.
12. S. M. Weiss and G. E. Kulikowski, Computer systems that learn, Kauffman, CA, 1991.

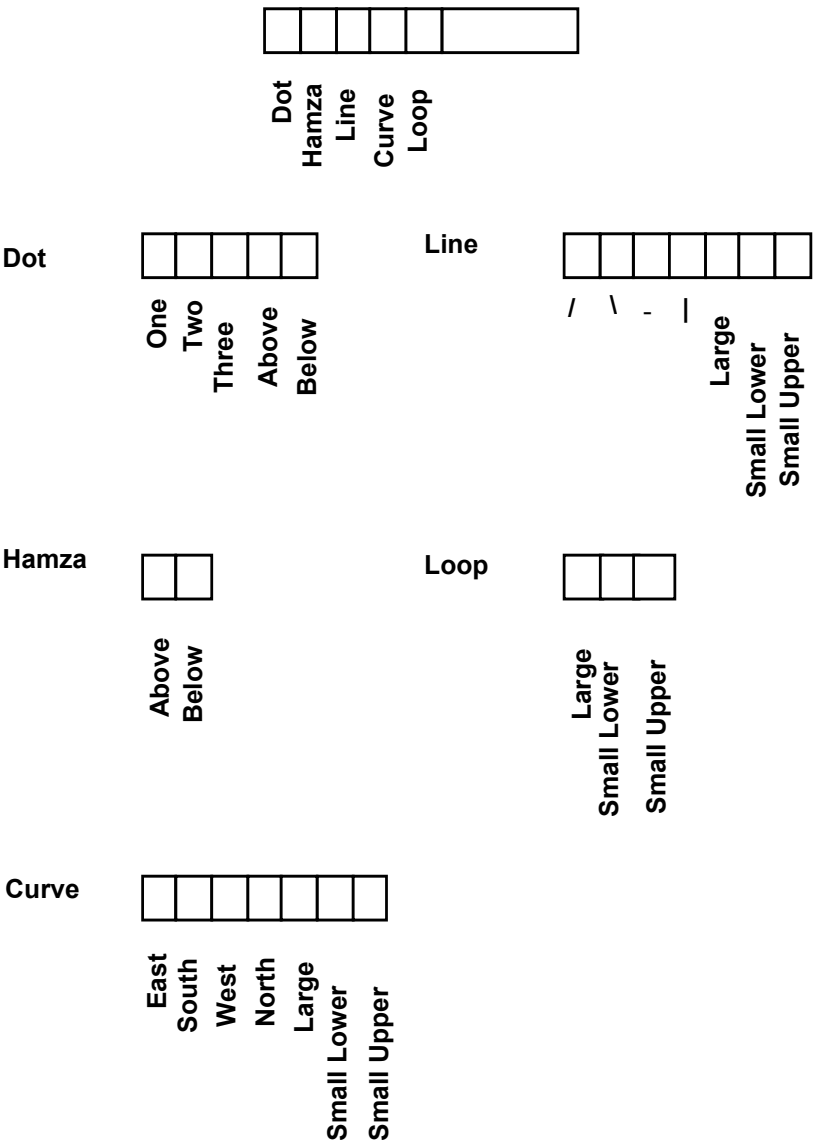

Fig. 3. Segment encoding for C4.5 input layer (each square cell holds a bit which acts as a flag)