Available online at www.sciencedirect.com
SCIENCE DIRECT®
Engineering Applications of Artificial Intelligence 18 (2005) 963–972
Engineering Applications of
ARTIFICIAL
INTELLIGENCE
www.elsevier.com/locate/engappai

A hybrid method for robust car plate character recognition

Xiang Pan*, Xiuzi Ye, Sanyuan Zhang

College of Computer Science, State Key Lab of CAD&CG, Zhejiang University, 310027 Hangzhou, PR China

Received 5 June 2004; accepted 24 March 2005
Available online 23 May 2005

Abstract

Image-based car plate recognition is an indispensable part of an intelligent traffic system. The quality of the images taken for car plates, especially for Chinese car plates, however, may sometimes be very poor, due to the operating conditions and distortion because of poor photographical environments. Furthermore, there exist some "similar" characters, such as "8" and "B", "7" and "T" and so on. They are less distinguishable because of noises and/or distortions. To achieve robust and high recognition performance, in this paper, a two-stage hybrid recognition system combining statistical and structural recognition methods is proposed. Car plate images are skew corrected and normalized before recognition. In the first stage, four statistical sub-classifiers recognize the input character independently, and the recognition results are combined using the Bayes method. If the output of the first stage contains characters that belong to prescribed sets of similarity characters, structure recognition method is used to further classify these character images: they are preprocessed once more, structure features are obtained from them and these structure features are fed into a decision tree classifier. Finally, genetic algorithm is employed to achieve optimum system parameters. Experiments show that our recognition system is very efficient and robust. As part of an intelligent traffic system, the system has been in successful commercial use.

Keywords: Car plate character recognition; Similar character recognition; Statistical classification; Structural classification; Multiple classifiers combination; Genetic algorithm

1. Introduction

In recent years, intelligent traffic control systems have been widely used in toll-pay, real-time monitoring and parking systems (Kato et al., 2002). Image-based car plate recognition is an indispensable part of such systems.

In general, the following three major steps are used in image-based car plate recognition systems, namely (1) locating of car plate regions; (2) segmentation of characters from the plate regions; and (3) recognition of car plate characters (Takashi et al., 2000). A number of methods have been developed to locate car plate regions from the images and to segment each character (Cui et al., 1997; Yu et al., 1997; Kim et al., 2002). However, few researches have been done for recognition of car plate characters. Neural network method has been employed to recognize car plate characters (Koval et al., 2003). The method can achieve promising performance if the quality of the given car plate image is well. However, the quality of image taken for car plates, especially for Chinese car plates, is not always well. This is due to the operating conditions (e.g., dust on the car plates) and distortion and/or degraded because of poor photographical environments. Furthermore, there exist some "similar" characters, such as "8" and "B", "7" and "T" and so on. They are less distinguishable because of noises and/or distortions. Experiments have shown that it is difficult to achieve high car plate recognition rates only by neural network method.

Recently, many methods have been developed to improve the character recognition rates, noticeably

*Corresponding author.
E-mail address: panxiangid@yahoo.com (X. Pan).

0952-1976/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.engappai.2005.03.011

multiple classifiers combination methods, such as multi-stage classification schemes (Ahmad and Shridhar, 1995), "parallel" combination of multiple classifiers (Xu et al., 1992; Huang and Suen, 1995; Kang and Kim, 1997). In addition, many researchers turned their attention to integrate two kinds of schemes (Zhang and Chen, 2002). Different classifiers with different features can complement each other, and group decision can reduce errors drastically and achieve higher recognition rates (Kittler et al., 1998).

In combining multiple classifiers, the main problem is how to integrate statistical and structural methods due to their strong complement. In the statistical approach, the input pattern is characterized by a set of N features, and its description is achieved by means of a feature vector belonging to N-dimensional space. Typically, the classification stage can be viewed as a statistical decision process, whose advantage is not sensitive to noises (Foggia et al., 1997; Anil et al., 2000). Meanwhile some small differences between similar patterns may be difficult to distinguish and therefore similar characters may be hard to recognize. Differing from statistical method, structural methods decompose the patterns into simpler primitives and obtain the properties of primitives and/or their relationships. Structural methods, on the other hand, are closer to human recognition strategy (Bunke and Sanfeliu, 1990; Pavlidis, 2003), and are more powerful to recognize similar patterns. However, the structural features are sensitive to noises and therefore difficult to be extracted. Therefore, how to combine statistical and structural methods together is crucial to achieve high recognition rates. Preliminary work along this line has been exploited, e.g., in structural/statistical feature-based vector (SSFBV) method (Heutte et al., 1998) by lumping statistical features and structural attributes into a vector as input to a statistical classifier.

In this paper, we propose a two-stage car plate character recognition system combining statistical and structural methods. Car plate image is skew corrected and normalized before recognition. In the first stage, four statistical sub-classifiers recognize the character images independently, and the recognition results are combined using the Bayes method. If the output of the first stage contains characters that belong to prescribed sets of similarity characters, structure recognition method is used to further classify: the character image is preprocessed once more, structure features are obtained and fed into a decision tree classifier. Finally, genetic algorithm (GA) is presented to deal with the problem of defining the system parameters. Unlike the SSFBV method that takes structural features as part of the input to the statistical classifier, our method is a two-step strategy, which performs the statistical recognition first, and then the structural recognition to those characters which have ambiguities. Furthermore, most parameters of system are obtained by GA instead of experience values, which make our system more stable. Our method has been integrated into a commercial car plate recognition system to replace the neural network method originally implemented in that system. We tested the method by experimental data as well as in real-world applications. Examples show that our method is very efficient and robust (See Section 7 for results and comparisons).

Although the proposed method deals with character recognition in car plates, which is part of video-based text recognition, our method can have potential applications in video retrieval and indexing, and in other related areas of video information processing.

The paper is organized as follows: Section 2 provides the overview of our car plate character recognition. Section 3 introduces methods for preprocessing character images, including skew correction and normalization. In Section 4, methods for recognizing characters by statistical approaches are described. In Section 5, structural methods are used to recognize similarity sets of characters. Section 6 discusses how to use GA to achieve optimal parameters of the system. Section 7 shows some experiment results by the proposed methods, and comparison with the results by original BP neural network schema is presented. Section 8 concludes the paper and gives some future work.

2. Overview of the system

In image-based car plate recognition system, each step plays an important role for high-performance system. Some steps, such as location of car plate regions and segmentation of characters from the plate regions, have not been presented in this paper (although we also develop excellent strategies for such steps). This paper focuses on how to recognize car plate characters accurately, especially such degraded and similar characters. Fig. 1 gives the architecture of our car plate character recognition system.

Our system can be decomposed into two stages. In the first stage, the combination of sub-classifiers' result is to make the system achieve high recognition performance for degraded characters because these sub-classifiers use different statistical features. After the recognition of the first stage, characters recognized incorrectly are mostly such similar characters. As a complement to the first stage, the second stage is designed to distinguish similar characters by structural methods. In experiment (See Section 7), we give some characters (degraded or similar characters) that are usually recognized incorrectly by neural network method, but can be recognized correctly by our hybrid system.

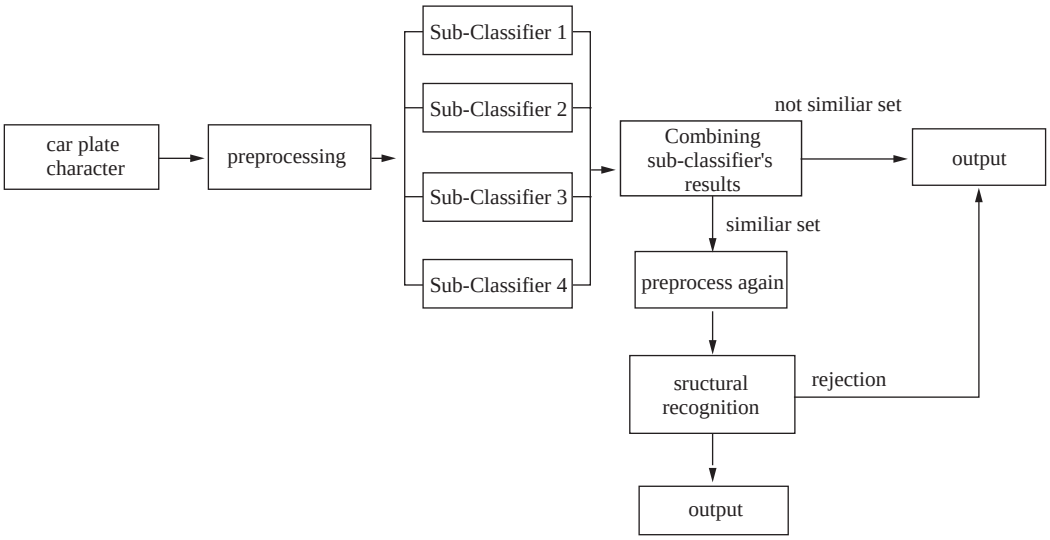

Fig. 1. Architecture of our car plate character recognition system.

3. Preprocessing

In order to recognize characters accurately, preprocessing the images, such as skew correction and normalization, have to be performed. In this section, we briefly introduce skew correction and normalization operations.

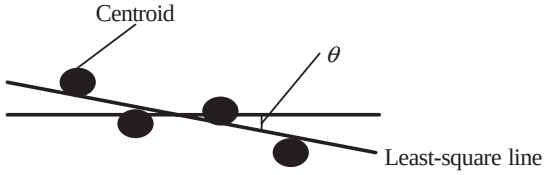

Fig. 2. Skew correction of car plate.

3.1. Skew correction

Character recognitions are generally very sensitive to skew. Therefore, skew detection and correction are critical. We propose here a least-square-based skew detection method. Suppose the binary image $F = \{F(i,j),\ i = 1,2,\ldots,I,\ j = 1,2,\ldots,J\}$ is defined as follows:

$$F(i,j) = \begin{cases} 0 & \text{white}, \\ 1 & \text{black}. \end{cases}$$

The main idea of our skew algorithm is to find "valid" character regions, and compute centroid of each "valid" region. The least-square line $y = kx + b$ is generated by these centroids. Finally, the skew angle θ between least-square line and horizontal line can be calculated by the equation $\theta = a\tan(k)$. Fig. 2 gives a simple description of skew correction.

The following is a detailed process of the skew correction algorithm.

Step 1: Find out all the connected regions. Let the connected region sets be $\{C_1, C_2, \ldots, C_N\}$, and C_i has a height H_i and width W_i.

Step 2: For each connected region, check if it is a "valid" region. A connected region C_i is said to be "valid" if $T_{\min} < W_i/H_i < T_{\max}$, where $T_{\min}$ and $T_{\max}$ are predefined values. For a given Chinese car plate, in general, the rate between width and height of each character ranges from 0.4 to 0.8. Therefore, in our implementation, $T_{\min}$ and $T_{\max}$ are set to 0.3 and 1.0, respectively.

Step 3: For each "valid" connected region, calculate its centroid (G_{ix}, G_{iy}):

$$G_{ix} = \frac{\sum_{(x,y)\in C_i} xC_i(x,y)}{\sum_{(x,y)\in C_i} C_i(x,y)},$$
$$G_{iy} = \frac{\sum_{(x,y)\in C_i} yC_i(x,y)}{\sum_{(x,y)\in C_i} C_i(x,y)}. \quad (1)$$

Step 4: Generate least-square line based on the centroid (G_{ix}, G_{iy}), and achieve skew angle θ. Given that $F(x,y)$ is the skew image and $F(x',y')$ the corrected image, the skew correction equation are defined as follows:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}. \quad (2)$$

Fig. 3 gives some images before and after skew correction.

After skew correction of the character images, characters are segmented from the corrected images.

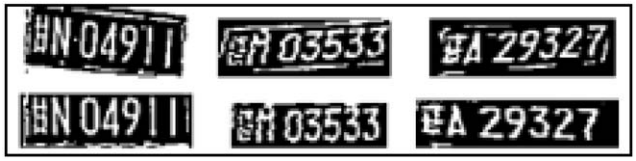

Fig. 3. Some images before (upper) and after (lower) skew correction.

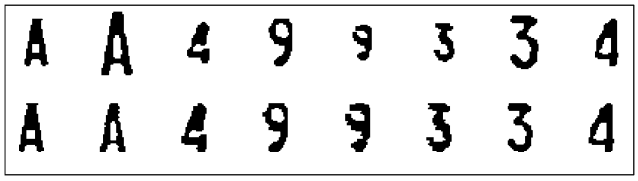

Fig. 4. Character images before (upper) and after (lower) size normalization.

3.2. Size normalization

Characters segmented from different car plates have different sizes. A linear normalization algorithm is applied to the input image to adjust to a uniform size (in our implementation, 32×16). Assume the horizontal and vertical projections of the original image F be H and V, respectively. The normalization position (m, n) of (i, j) is obtained by

$$m = \sum_{k=1}^{i} H(k) \times \frac{M}{\sum_{k=1}^{I} H(k)},$$
$$n = \sum_{k=1}^{j} V(k) \times \frac{N}{\sum_{k=1}^{J} V(k)}, \quad (3)$$

where M, N is the height and width of normalized image.

Fig. 4 shows some normalization results:

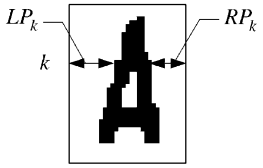

Fig. 5. The left–right contour feature.

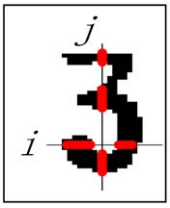

Fig. 6. The line segment features (the count of line segment is 3 in jth column and 2 in ith row).

4. Recognizing characters by statistical methods

After preprocessing, the input character image is first recognized by statistical methods. In our approach, four sub-classifiers recognize the character independently, and their recognition results are combined using the Bayes method.

4.1. Sub-classifiers

Four sub-classifiers are designed independently, and different features are utilized. The recognized method for sub-classifiers is template matching due to its efficiency and stability. We give a brief description about these features used in the system.

Sub-classifier 1: Sub-classifier 1 uses the zoning density (Trier et al., 1996). To calculate the zoning density, the input image is divided into row × col ($\text{row} = 4, \text{col} = 4$) zones. In each zone, the density of black points is calculated. Thus the feature has 32 components for the input character image.

Sub-classifier 2: The input feature is the vertical projection $\{y(i) \mid i = 1, 2, \ldots, 16\}$, where $y(i)$ is the number of black pixels of the ith column.

Sub-classifier 3: The input feature of this sub-classifier is the left–right contour feature. The contour feature can be obtained as follows: in kth row, scan the image from left to right boundary. Whenever the pixel turns out to be black, compute the width LP_k between the pixel and left boundary. The width RP_k between the pixel and right boundaries can be similarly calculated. Fig. 5 shows the left–right contour feature (to be clear enough, the original image has been expanded). There exist 64-dimension left–right contour features.

Sub-classifier 4: In sub-classifier 4, we extract line segment features of the input character. For each row, we count the number of line segment. The same operation is performed for each column. Therefore, the line segment features have 48 dimensions (the image size is 32∗16). Fig. 6 gives an example of such features.

4.2. Combining sub-classifiers using Bayes method

To achieve robustness and high recognition rates in the recognition system, the recognition results from the above four sub-classifiers are combined by the Bayes method (Xu et al., 1992).

Suppose that the pattern space P consist of M disjoint sets

$$P = C_1 \cup C_2 \cdots \cup C_M,$$

where each C_i, $\forall i \in \{1,2,\ldots,M\}$ represents a pattern class (in our implementation, $M = 34$). A sub-classifier e_k can be considered as a black box, whose input is the extracted feature from the given character image and whose output is the classify result.

The recognition error of the kth classifier is defined in the form of its confusion matrix as follows:

$$PT_k = \begin{Bmatrix} n_{11}^{(k)} & n_{12}^{(k)} & \cdots & n_{1M}^{(k)} \\ n_{21}^{(k)} & n_{22}^{(k)} & \cdots & n_{2M}^{(k)} \\ \vdots & \vdots & \ddots & \vdots \\ n_{M1}^{(k)} & n_{M2}^{(k)} & \cdots & n_{MM}^{(k)} \end{Bmatrix}$$

for $k = 1,2,\ldots,K$; where each row i corresponds to the class C_i and each column j corresponds to the event $e_k(x) = j$. Thus, an element $n_{ij}^{(k)}$ denotes that $n_{ij}^{(k)}$ samples of class C_i have been assigned a label e_k.

The confusion matrix can be obtained by classifying a testing set and stored as a prior knowledge. In other words, it can be regarded as an expert who shows the belief that $e_k(x) = j$. The following equation is used to compute the belief:

$$P(x \in C_i / e_k(x) = j) = \frac{n_{ij}^{(k)}}{n_j^{(k)}} = \frac{n_{ij}^{(k)}}{\sum_{i=1}^{M} n_{ij}^{(k)}}, \quad i = 1,\ldots,M. \tag{4}$$

For all K sub-classifiers $e_1, \cdots e_k \cdots e_K$, K confusion matrixes $PT_1,\ldots,PT_K$ are constructed. Giving input character image F, K sub-classifiers have K decisions $e_k(x) = j_k$, $k = 1,\ldots,K$. The aim of combination is to compute the probability that a sample belongs to each of the M classes with K decisions and K confusion matrixes.

Several rules, such as sum rule, product rule, max rule, min rule (Kittler, 1998), can be used to combine different classifier results. We compare recognition performance based on different combination rules and experiment shows product rule can achieve better results. Therefore, product rule is employed in our system. For a given input character F, the probability that F belongs to each class can be computed by the following equation:

$$\mathrm{bel}(i) = \eta \prod_{k=1}^{K} P(x \in C_i \mid e_k(x) = j_k), \tag{5}$$

where η is a constant that ensures $\sum_{i=1}^{M} \mathrm{bel}(i) = 1$. That is to say,

$$\frac{1}{\eta} = \sum_{i=1}^{M} \prod_{k=1}^{K} P(x \in C_i / e_k(x) = j_k). \tag{6}$$

At this point, we are able to obtain the set $\mathrm{bel}(i)|_{i=1,2,\ldots,M}$ and sort them in the descending order. We assume their corresponding class-type sequence to be $\{P_1, P_2, \ldots, P_M\}$. If the subset $\{P_1, P_2\}$ does not belong to the prescribed sets of similarity characters or $|\mathrm{bel}(1) - \mathrm{bel}(2)| > T_{\mathrm{gate}}$, the class P_1 is taken as the final recognition result. Otherwise, the character is further recognized by the structural methods described below in Section 4 where T_{gate} is a predefined parameters, and how to define it will be described in Section 6.

5. Recognizing similar characters by structural methods

To recognize similar characters in our car plate character recognition system, it is important to extract stable and representative structure features. Fortunately, different similarity sets have different structural features. We will discuss in this section structure features to distinguish most frequently occurring similarities: "8" and "B" by using left-edge contour feature.

5.1. Edge contour smoothing and contour feature extraction

The left edges of "B" and "8", ideally, are a straight line and a curved line, respectively. However, more than often, noisy line segments exist in the left edges. Such noisy line segments can seriously affect the extraction of the edge contour features. Unfortunately, traditional smoothing algorithms, such as the one by Unger (1959), cannot remove these noises.

Considering images in Fig. 5 (upper) of "8" and "B", we find structural features of character "8" are not affective to noises. However, the character of "B" with noises may appear to be an "8". We propose here a special smoothing method based on left edge information to remove these noisy line segments.

Definition 1. A point (i, j) with

$$F(i,j) \cap \left[\bigcup_{0 \leqslant u \leqslant 1, 0 \leqslant v \leqslant 1} \overline{F(i+u, j+v)} \right] = 1$$

is called an edge point. An edge point can be classified further into left, right, top and bottom edge points if $\mathrm{point}(i,j-1) = 0$, $\mathrm{point}(i,j+1) = 0$, $\mathrm{point}(i-1,j) = 0$, $\mathrm{point}(i+1,j) = 0$.

The following describes the edge contour-smoothing algorithm.

Step 1: Compute total black point black_k and left edge point black_k of each column k. In addition, record the beginning position $\mathrm{begin}_{k,l}$ and length $\mathrm{Len}_{k,l}$ of each vertical continuous run that in kth column.

Step 2: Smooth left edge contour. For each column, the line segment satisfying condition $\mathrm{Len}_{k,l} < TW1$ is removed first. Furthermore, if the kth column satisfies condition $\mathrm{left}_k > \frac{1}{2} T_{\mathrm{black}}$, all black points are removed in the left of the kth column are removed, where T_{black} and

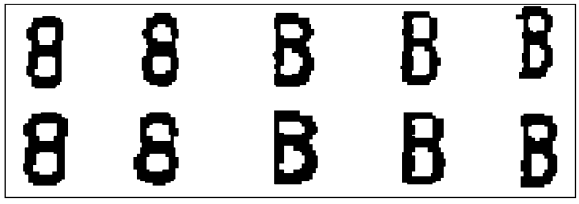

Fig. 7. Similarity set "8" and "B" before (upper) and after (lower) smoothing.

$TW1$ are predefined threshold. The first condition is used to remove small noisy line segments, and the second means that the given character is mostly like "B".

Fig. 7 shows the effect of contour smoothing algorithm. Notice that for the characters "8", little effect can be seen due to the fact that its left is a curved line. The effect, however, can be observed for character "B".

After edge smoothing, the task is left to extract structure feature. First, the left edge sequence of the input image F is defined, which is a left edge point set $\{F(i,k_i) \mid i = 1,2,\ldots,M\}$. For point $F(i,k_i)$, the value k_i can be obtained by the following process: in the ith row, the column index j moves from left to right until $F(i,j)$ is a left edge point, and $k_i = j$ (the last value). Then the curve direction of edge point (i,j) is defined as follows:

$$\mathrm{dir}_i = \begin{cases} 1 & f(i,j) - f(i-1,j) > 0, \\ 0 & f(i,j) - f(i-1,j) = 0, \\ -1 & f(i,j) - f(i-1,j) < 0. \end{cases} \quad (7)$$

Let the curve direction sequence be $\{\mathrm{dir}_i \mid i = 1,\ldots,M\}$, if dir_i and dir_k satisfy the following conditions:

$$\mathrm{dir}_i \times \mathrm{dir}_k > 0; \quad \mathrm{dir}_t = 0 \mid t = {}_{t=i+1,\ldots,k-1}; \quad \|i-k\| < W. \quad (8)$$

Then the sequence is said to contain *a curve point*, where W is a predefined threshold.

The left edge contour feature is calculated as follows:

Step 1: Obtain the left edge sequence $\{F(i,k_i) \mid i = 1,2,\ldots,M\}$ of the input image F.

Step 2: Compute the curve direction of the left edge sequence $\{\mathrm{dir}_i \mid i = 1,\ldots,M\}$.

Step 3: Compute the total of the curve point set (denoted by totalcurve) from the direction of the left edge sequence.

Step 4: Approximate the left edge sequence by using a least-square method. Compute the approximate error (denoted by error).

The two types of structural features are fed into a binary decision tree to distinguish "8" and "B".

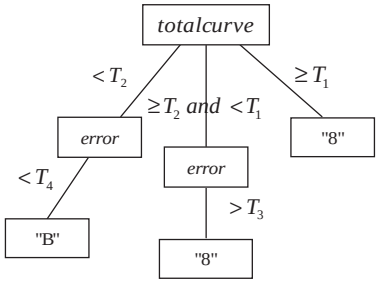

Fig. 8. Binary decision tree to recognize the similar characters "8" and "B".

5.2. Structural recognition by binary decision tree classifier

After extracting structural features, how to design classifier to recognize similar characters is another problem. Binary decision trees are served as structural classifiers. There are many methods used for generating decision tree automatically, such as C4.5 (Quinlan, 1993; Gelfand et al., 1991). These methods, however, are not suitable for our systems because these methods presume the parameters used for structural feature extraction have been predefined. In our system, many parameters (such as T_{black}) used in structural features remain undecided, and their values are very important for system performance. Therefore, we simply build decision tree by the following process: different structure features are flagged with different importance, and sorted in the descending order of their importance. First the most important feature is set to be root node. Then the second important feature is served as the second layer node, and so on. The importance of different structure features can be judged by the experiment. Take for example, to differentiate "8" and "B", the curve point is the most important feature and is set to root node. Fig. 8 shows such a decision tree, where T_1, T_2, T_3, T_4 are decision parameters. After deciding the tree structure, the crucial problem is how to achieve optimal system parameters, which is discussed in Section 4.

The decision tree does not always give the precise result. If the decision rejects the character, the final recognize result is set back to P_1 (refer to Section 4).

6. Parameters optimization by genetic algorithm

In our system, several parameters (such as $T_{\mathrm{gate}}, W, T_1, T_2, \ldots$) need to be predefined and the recognition performance is fine only if they are set at suitable values. As a consequence, how to decide them is a very important but difficult problem. This section discusses how to use GA to achieve optimal parameters.

GA has mainly been used as function optimization, which has been shown to be effective global optimiza-

tion tools. The basic concept of GA is the survival of the fittest and natural selection described by Goldberg (1989). In GA, each possible solution is usually coded as a binary string, and the algorithm finds the best values by some genetic operators such as reproduction, crossover and mutation.

The optimization purpose is to achieve suitable parameters for higher recognition performance. We therefore use the recognition rate of the given testing characters as fitness function. Too much testing characters, however, will lead to a very slow convergence speed, even to premature convergence. To speed up the convergence of the GA, the testing character sets are designed by the following process: for a pair of similar characters, the testing character set only contains two kinds of such similar characters. So we need perform several GAs to obtain system parameters. Take for an example, if we want to decide parameters for similar characters {"8", "B"}, the testing characters consist of characters belonging to set {"8", "B"}. The following is a summary of GA.

Initialization: This step defines the ranges and precision of parameters. Also we need to define the total genetic generations generation and population size popsize. They are set to be 100 and 20 in our implementation.

Coding: Each parameter is coded into a binary substring and all sub-strings are connected as a string called chromosome, denoted by S_i. Each bit of S_i is randomly set to 0 or 1.

Fitness calculation and scaling: Final recognition rate of the given testing characters is computed and served as Fitness f_i. We compute maximal fitness $f_{\max} = \max(f_i)$, $i = 1, 2, \ldots$ popsize, and average fitness $f_{\text{avg}} = \sum f_i/\text{popsize}$. The two values are useful in mutation process.

Reproduction: First, the best string $S_{\max}$ (the string owns the best fitness) is copied to the next generation, and other strings S_i are copied with the probability of $p_i = f_i/\sum f_i$.

Crossover: All strings are paired randomly. Based on the chosen probability p_c ($p_c = 0.6$, a typical value in genetic algorithm), a subset of these pairs experience crossover, and some new strings are generated.

Mutation: In order to generate new population values, bits of the resultant strings are randomly changed according to probability p_m. To avoid the premature convergence, the value p_m is defined by the following equation:

$$p_m = \begin{cases} 0.015 & 0.4 \leqslant m < 0.5, \\ 0.02 & 0.3 \leqslant m < 0.4, \\ 0.03 & 0.2 \leqslant m < 0.3, \\ 0.035 & 0.1 \leqslant m < 0.2, \\ 0.04 & m < 0.1, \end{cases} \quad (9)$$

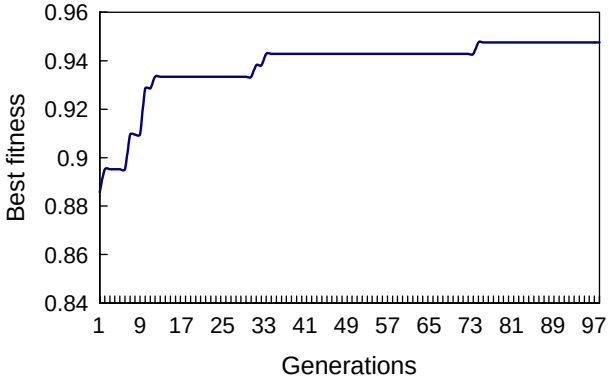

Fig. 9. The genetic process for similar characters "8" and "B".

where $m = (f_{\max} - f_{\text{avg}})/f_{\max}$ is called density factor. The parameter, as it stands, represents the density degree of current generation. If the value m is small, the current generation has possibility of premature convergence. So the mutation probability p_m needs to be set a bigger value.

For other similar characters, we can obtain parameters using analogical process. Notes the threshold T_{gate} is different for different similar characters. Fig. 9 gives the generic process for similar characters "8" and "B".

7. Experimental results

7.1. The character image database

For our experiments, we have built a car plate character image database with images taken from different locations, types of cars, road conditions, illumination and cleanness conditions. It has 34 types of characters, including 24 uppercase and 10 numbers,[1] the maximum sample number for training and testing is 50. Fig. 10 shows some color car plate images extracted from the original image.

7.2. Recognition result

To test the performance of the hybrid method, we compare the results from our method with the Bp neural network (MultiLayer Perception with back propagation training algorithm) which had been already implemented in the original car plate recognition system and had been already in commercial use.

[1] In Chinese car plate, "0" and "O" use the same prototype, and can be judged by the position. The same for "1" and "I".

Fig. 10. Sample car plate character images from our image database.

Table 1
The recognition rates of our hybrid method vs. the Bp neural network method

Method	Training set (%)	Testing set (%)
Bp neural network	95.68	91.03
Our hybrid method	97.46	95.41

The Bp neural network is similar to the SSFBV method: several kinds of features are lumped into a vector as input of the Bp neural network. The Bp neural network is a three-layer structure, and both hidden layer have 50 nodes.

The recognition rates listed in the following table (without considering the rejected characters for which no decisions will be made for the characters).

It can be seen from Table 1 that our method has small improvement on the recognition rate over the Bp neural network method. However, the difference for testing character set is very obvious. Furthermore, the performance of Bp neural network has huge difference for training and testing set. Our method, however, shows more stability.

To further prove the robustness and efficiency of our hybrid method, we tested more than 10 000 car plate images taken from real world application. As shown in the graph in Fig. 11, the recognition rates differ greatly with the road conditions, illumination and cleanliness of cars. Compared with Bp neural, our hybrid method can always achieve a drastic improvement in the recognition performance. (The Chinese characters in the first location of the car plate images are excluded from the recognition.)

Fig. 12(a) shows some typical car plate images recognized incorrectly by Bp neural network (characters recognized incorrectly are flagged with red frames). However, these car plate images can be recognized correctly by our hybrid method.

Our hybrid method has been applied into real-time car plate recognition system. The captured image resolution is 240×320, and the recognition time is about 0.1–0.2 s for each car plate image. Therefore, the system will work well if cars run with the speed under 100 km/h. Now the system is wildly applied in different Chinese geographical provinces, and proven to be robust and stable. While our system is not perfect, and it still exists some shortcomings. For example, the system cannot achieve very wonderful recognition performance in high-speed road in which cars' speed will be more than 120 km/h.

8. Conclusion and future work

In this paper, we propose a two-stage hybrid method for car plate character image recognition. The main contributions of our work include (1) Distinguishing similar characters by local structural features. (2) Developing a system architecture combining statistical and structural recognition methods. (3) Determining system parameters by a genetic algorithm. We tested the method with large number of plate images captured in different environments from real applications. The method has been successfully used in commercial car plate recognition systems. Compared with Bp neural network, our hybrid method is more effective and robust.

Our current work is focused on improving the recognition performance to make the system work well for high-speed highway applications. Our future work includes how to recognize the Chinese characters presented in the Chinese car plates, and how to extend our method in the area of video text analysis and processing.

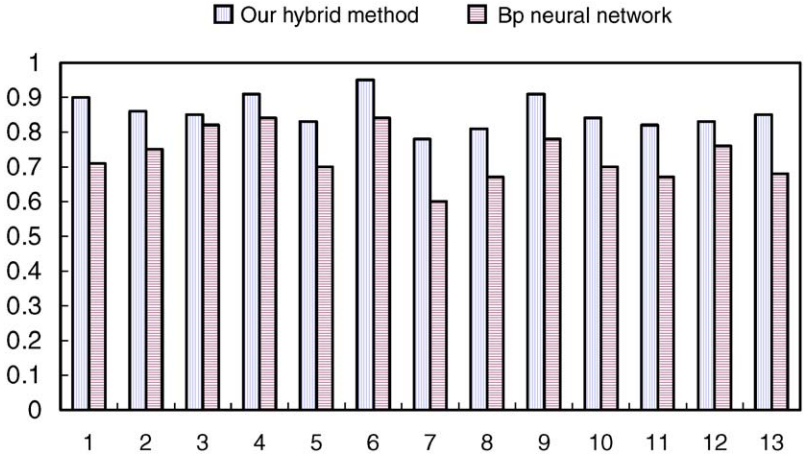

Fig. 11. Comparison of recognition rates of our hybrid method and the Bp neural network for more than 10 000 car plate character images in 13 Chinese geographical provinces.

Fig. 12. Improved recognition results by our hybrid method over the Bp neural network. (a) The images and the incorrect recognition results by Bp neural network. (b) The same images and the correct recognition results by our hybrid method.

Acknowledgements

The authors gratefully acknowledge Prof Ching Y. Suen of the Concordia University, Canada, for his suggestive comments. The authors would like to thank the support from the Cheung Kong Scholar's Program Funds at Zhejiang University, China NSF under Grant (#60273060,60473106), and Chinese Ministry of Science and Technology under Grant #2003AA4Z3120, and Nanchang Li De Feng Technology Co. Ltd. Thanks are also given to the anonymous reviewers for their kind comments. Part of the work has been supported by the Education Office of Zhejiang Province under Grant #G20030433.

References

Ahmad, C.J., Shridhar, M., 1995. Recognition of handwritten numerals with multiple feature and multistage classifier. Pattern Recognition 28 (2), 153–160.

Anil, K., Duin, R.P.W., Mao, J., 2000. Statistical pattern recognition: a review. IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (1), 4–34.

Bunke, H., Sanfeliu, A., 1990. Syntactic and Structural Pattern Recognition, Theory and Applications. World Scientific, Singapore.

Cui, Y., Cui, Q.Y., Huang, Q., 1997. Automatic license extraction from moving vehicles. International Conference on Image Processing, pp. 126–129.

Foggia, P., Sansone, C., Tortorella, F. et al., 1997. Character recognition by geometrical moments on structural decomposition. International Conference on Document Analysis and Recognition, UlM, Germany, vol. 1, pp. 6–10.

Gelfand, S.B., Ravishankar, C.S., Delp, E.J., 1991. An iterative growing and pruning algorithm for classification tree design. IEEE Transactions on Pattern Analysis and Machine Intelligence 13 (2), 163–174.

Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, MA.

Heutte, L., Paquet, T., et al., 1998. A Structural/statistical feature based vector for handwritten character recognition. Pattern Recognition Letters 19, 629–641.

Huang, Y.S., Suen, C.Y., 1995. A method of combining multiple experts for the recognition of unconstrained handwritten numerals. IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (1), 90–93.

Kang, H.J., Kim, J., 1997. A probabilistic framework for combining multiple classifier at abstract level. Proceedings of the Fourth International Conference Document Analysis and Recognition, Ulm, Germany, August, vol. 1, pp. 870–874.

Kato, T., Ninomiya, Y., Masaki, I., 2002. Preceding vehicle recognition based on learning from sample images. IEEE Transactions on Intelligent Transportation Systems 3 (4), 252–260.

Kim, K.I., Jung, K., Kim, J., 2002. Color texture-based object detection: an application to license plate localization. International Workshop on Pattern Recognition with Support Vector Machines, pp. 293–309.

Kittler, J., Mohamad, H., Robert, P.W., 1998. On combining classifiers. IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (3), 226–239.

Koval, V., Turchenko, V., Kochan, V., 2003. Smart license plate recognition system based on imaging processing using neural network. IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, pp. 8–10.

Pavlidis, T., 2003. 36 years on the pattern recognition front. Pattern Recognition Letters 24, 1–7.

Quinlan, J.R., 1993. C4.5: Programs for Machine Learning. Morgan Kaufmann, San Mateo.

Takashi, N., Toshihiko, T., Keiichi, Y., et al., 2000. Robust license-plate recognition method for passing vehicles under outside environment. IEEE Transactions on Vehicular Technology 49 (6), 2309–2319.

Trier, O.D., Jain, A.K., et al., 1996. Feature extraction methods for character recognition—a survey. Pattern Recognition 29 (4), 641–661.

Unger, S.H., 1959. Pattern detection and recognition. Proceedings of the IRE, pp. 1737–1752.

Xu, L., Krzyzak, A., Suen, C.Y., 1992. Methods of combining multiple classifiers and their application to handwriting recognition. IEEE Transactions on System Man and Cybernetics (22), 418–435.

Zhang, P., Chen, L.H., 2002. A novel feature extraction method and hybrid tree classification for handwritten numeral recognition. Pattern Recognition Letters 23, 45–56.