International Journal of Computational Intelligence Systems, Vol. 7, Supplement 1 (2014), 113-120

Offline recognition of degraded numeral characters with MMTD-based fuzzy classifiers*

Weiqing Cheng† and Long Hong
College of Computer, Nanjing University of Posts and Telecommunications,
Nanjing, Jiangsu 210003, China
Jiangsu High Technology Research Key Laboratory for Wireless Sensor Networks,
Nanjing, Jiangsu 210003, China
State Key Laboratory of Software Development Environment, Beihang University,
Beijing, 100191, China

Shaobai Zhang
College of Computer, Nanjing University of Posts and Telecommunications,
Nanjing, Jiangsu 210003, China

Received 15 December 2012

Accepted 29 June 2013

Abstract

Enabling machines to read like human beings has been a hot issue for more than fifty years. A novel offline degraded numeral recognition method (DNRBM) based on the measure of medium truth degree (MMTD) is proposed in this paper to identify segmented degraded numeral characters in gray images. It consists of distinguishing foreground from background, rotating an image, wiping off mottles, cutting margins, calculating both statistic and structural features, and recognizing numerals by the fuzzy classifiers constructed based on MMTD using features selected by logistic regression. The experimental results show that in comparison with the template matching method and the k-NN method, the proposed method performs well on recognizing degraded numeral characters with better scalability and better recognition performance.

Keywords: the measure of medium truth degree (MMTD); classification based on MMTD (CBM); logistic regression; feature selection; offline recognition of degraded numerals.

1. Introduction

Offline recognition of numerals has many practical applications such as automatic number plate recognition and zip code recognition[1]. As a classical and challenging problem, it has been studied for many years. Many methods for offline recognition of English, Arabic (Indian), and Persian printed or handwritten numeral characters have been proposed[1-10]. However, only a few of them have taken degraded English numerals into account[4-6]. Besides, most methods need a large number of samples for training, which are not always obtainable. This paper aims to address degraded english numeral recognition using proposed fuzzy classifiers fit for a small quantity of training samples, and to test the classification performance of the proposed fuzzy classifiers by taking the numeral recognition as an example. Numeral recognition usually includes such processes as preprocessing on input images, feature extraction, feature selection, classification based on different classifiers, such as HMM (Hidden Markov Model), SVM (Support Vector Machines), k-NN (k-Nearest Neighbours), *etc.*, and postprocessing that refines and improves recognition results[1]. In this paper an integrated method called DNRBM is proposed to identify segmented degraded

*Corresponding author: chengweiq@njupt.edu.cn
†College of Computer, Nanjing University of Posts and Telecommunications

numeral characters in gray images. The method consists of such steps as distinguishing foreground color from background color, rotating a monochrome image, wiping off mottles in an image, cutting margins of an image, calculating features, and recognizing numerals by the fuzzy classifiers that are established based on the measure of medium truth degree (MMTD)[11-12] and several features selected by an embedded feature selection approach using the logistic regression.

The remaining of the paper is organized as follows. Features are presented and feature selection is addressed in Section 2, where two types of features are used. A classification approach based on MMTD (CBM) to recognize degraded numeral characters (DNRBM) is proposed in Section 3. In Section 4, experiments are conducted to verify the performance of the proposed recognition method, where the MMTD based classifiers are compared to k-NN classifiers based on the same training and test sets since k-NN classifiers usually achieve high performance. Finally, the conclusions are presented in Section 5.

2. Feature Definition and Feature Selection

Features are the information extracted from the image of a word or character, and they are used to build classifiers for classification[1]. In this paper, we assume that there is only one degraded character in a gray image and the image size and the character size are variable. One challenge in feature extraction from a degraded character in a gray image is to locate the character therein. The other challenge is to define features that are dependent on numerals and robust, such as independent of the character size and having almost invariable values for the same numeral in images with different degrees of clarity, as well as to determine which features are more suitable for classification.

2.1. *Distinguishing between foreground and background*

To determine the character location in a gray image, the image should be firstly converted to a black-and-white image. The pixels in the foreground are generally less than those in the background; thus we design the following algorithm to distinguish between foreground and background:

1) Read the gray image, and store the pixels in the array *pixs* with *pixs*(0,0) and *pixs*(*height*-1,*width*-1) denoting the pixel at the top left corner and the pixel at the bottom right corner;

2) Get the frequency (i.e. number of pixels) for each color value in the image;

3) Find the color with the maximum number of pixels, denoted as c_1; if the image has only this color then return error;

4) Try to find the color with the maximum number of pixels, denoted as c_2, among colors with the gaps between those and c_1 bigger than a threshold, e.g. the total number of all colors (e.g. 256 for 8 bit pixel) divided by 4; if there is no such color, then try to find the color with the maximum number of pixels among colors except c_1, also denoted as c_2;

5) Count the number of pixels on the borders with their colors nearer to c_2 than to c_1; if the number is greater than 65% of the total number of pixels on the borders, then swap c_1 and c_2;

6) If $c_1 > c_2$, then the pixels with their colors $\geq (c_1+c_2)/2$ are set to the specified background color (for example, white), and other pixels are set to the specified foreground color (for example, black); If $c_1 < c_2$, then the pixels with their colors $\leq (c_1+c_2)/2$ are set to the background color, and other pixels are set to the foreground color.

2.2. *Wiping off mottles*

The original gray image may be blurry, thus it is better to wipe off mottles in the derived monochrome image. The foreground of the monochrome image is regarded as a set of connected subimages. If a pixel and one of its eight neighbors are both in the foreground, then the two pixels are deemed connected. If pixel *A* is connected with pixel *B*, and pixels *B* and *C* are connected, then pixels *A* and *C* are also deemed connected. Any pixel in a connected subimage is not connected to any pixel outside the subimage.

We deem that the character in the image consists of one or few big connected subimages, while other connected subimages containing few pixels in the image are regarded as mottles. We design the following algorithm to wipe off mottles:

1) Find all connected subimages and count the pixel number of every subimage;

Starting at any foreground pixel *p*, the connected subimage containing the pixel *p* can be found by recursively inspecting each of its neighbor foreground pixels to find all foreground pixels connected with the pixel *p*.

2) A subimage with its pixel number less than a threshold, e.g. 7 percent of the total number of foreground pixels, is taken as a mottle;

3) A subimage with its pixel number less than a threshold, e.g. 25 percent of the pixel number of the biggest subimage, is taken as a mottle;

4) Wipe off mottles by setting the pixels in the mottles to the background color.

2.3. *Cutting margins*

This step is to cut margins of the new monochrome image so as to locate the numeral character therein. In fact, we do not actually cut the margins, and just need to know the coordinates for the top left corner and the bottom right corner of the foreground. This step includes the following processes:

1) Find the minimum rectangle containing all foreground pixels with the top left corner denoted as (*lowrow*, *lowcol*), and the bottom right corner denoted as (*highrow*, *highcol*);

2) If the width of the foreground is less than 6, then set *lowcol*=max(*lowcol*-2,0), *highcol*=min(*highcol*+2, *width*-1);

3) If *highcol*-*lowcol*+1 is still less than 6 or *highrow*-*lowrow*+1 is less than 6, then return error; else the subimage with the top left corner (*lowrow*, *lowcol*) and the bottom right corner (*highrow*, *highcol*) is *the normalized image* filled with a numeral character.

2.4. *Calculating features*

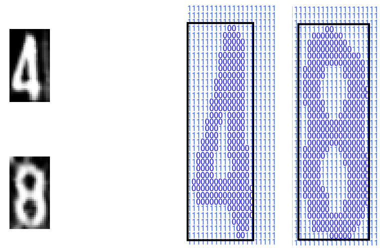
Figure 1 shows two original images, and the resulted images after processed by steps in Secs. 2.1~2.3. The images in two rectangles are the normalized images, whose features are to be calculated. Twenty-eight statistic features X_0~X_{26} and X_{28} as well as one structural feature X_{27} are defined in this paper. Each normalized image is logically divided into a 3×3 grid (three rows and three columns) averagely, as shown in Fig. 2. The fore 27 features and the feature X_{28} are defined based on the grid; all features are defined as follows:

$$X_i = \begin{cases} n_{r,c}/n_r, & \text{if } (i\%9)\%3 \text{ equals } 0 \\ n_{r,c}/n_c, & \text{if } (i\%9)\%3 \text{ equals } 1 \\ n_{r,c}/n_{all}, & \text{if } (i\%9)\%3 \text{ equals } 2 \end{cases} \tag{1}$$

$$X_{27} = \frac{\text{the height of the foreground}}{\text{the width of the foreground}} \tag{2}$$

$$X_{28} = n_r / n_{all}, r = 0 \tag{3}$$

where i=0,1,2,…,26, r=i/9, c=(i%9)/3, i/9 and i%9 mean the integral quotient and the arithmetical compliment of i divided by 9 respectively, $n_{r,c}$ denotes the number of foreground pixels in grid (r,c), n_r and n_c denote the number of foreground pixels in the row r and that in the column c respectively, and n_{all} denotes the total number of all foreground pixels.



(a) (b)

Fig. 1. Locating the character. (a) original images, (b) cleaned images

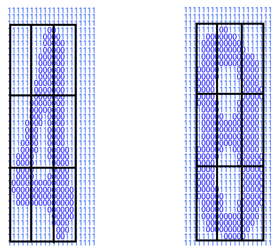


Fig. 2. Normalized images divided into 3×3 grids to calculate features

Which of the fore 27 features are used for classification depends on the results of feature selection, while X_{27} and X_{28} are directly used to construct fuzzy classifiers since they are intuitively distinguishing.

2.5. *Feature selection based on logistic regression*

Feature selection can help to provide faster and more efficient models so as to generate more reliable estimates by excluding noises and focusing on a smaller number of features[13]. We adopt an embedded feature selection approach using the logistic regression.

First, some numeral images are selected as training samples, and the fore 27 features for each sample are calculated according to Eq. (1). In this study, 65 training samples with 6 or 7 ones for each numeral are used for feature selection. Next, let X_0~X_{26} be the independent variables, and Y that takes values 0,1,2,..,9 be the response variable; construct a total of the number of response levels (denoted as c) minus 1 cumulative logistic regression models

$$\operatorname{logit}\left(\sum_{i=0}^{k}\pi_i\right) = \ln\left(\frac{\sum_{i=0}^{k}\pi_i}{1-\sum_{i=0}^{k}\pi_i}\right) = \alpha_k + \sum_{j=0}^{m-1}\beta_j X_j, \quad (4)$$

where m=27, c=10, k=0,1,2,…,c-2, i denotes a response level, m denotes the number of features, c denotes the number of response levels or classes, $\pi_i = P(Y = i|X)$, $\sum_{i=0}^{c-1}\pi_i = 1$, $\alpha_0,\alpha_1,\ldots,\alpha_{c\text{-}2}$ are intercept parameters, β_j is termed a logistic regression coefficient, X_j is an independent variable, and logit is a dependent variable.

Finally, let alpha be 0.05, and then maximum likelihood estimate of the coefficient of a feature having a p-value of 0.05 or less is considered to be statistically significant. Each feature with statistically significant parameter estimate should be selected. In this study, features selected from X_0~X_{26} consist of X_0~X_3, X_5~X_8, X_{11}, X_{14}, X_{16}, X_{17}, X_{20}, X_{23}, and X_{26}.

3. Classification based on MMTD

We propose a classification approach based on MMTD (the measure of medium truth degree[11]) (CBM) to recognize degraded numeral characters (DNRBM).

3.1. *Measure of medium truth degree*

Definition 1. Assume f_1: $X\rightarrow\boldsymbol{R}$, f_2: $X\rightarrow\boldsymbol{R}$ are two types of one-dimensional quantification mappings for set X of objects. Let $P(x)$ be a predicate, and two distance ratio functions relative to P, that is, h_{T1}: $f_1(X)\rightarrow\boldsymbol{R}$ and h_{T2}: $f_2(X)\rightarrow\boldsymbol{R}$ are defined as below:

$$h_{T1}(y) = \begin{cases} \dfrac{-2.0d(y,\alpha_F-\varepsilon_F)}{d(\alpha_T-\varepsilon_T,\alpha_F-\varepsilon_F)}, & y < \alpha_F-\varepsilon_F \\ 0, & \alpha_F-\varepsilon_F \le y \le \alpha_F+\varepsilon_F \\ \dfrac{d(y,\alpha_F+\varepsilon_F)}{d(\alpha_T-\varepsilon_T,\alpha_F+\varepsilon_F)}, & \alpha_F+\varepsilon_F < y < \alpha_T-\varepsilon_T \\ 1, & \alpha_T-\varepsilon_T \le y \le \alpha_T+\varepsilon_T \\ \dfrac{d(y,\alpha_F+\varepsilon_F)}{d(\alpha_T+\varepsilon_T,\alpha_F+\varepsilon_F)}, & y > \alpha_T+\varepsilon_T \end{cases} \quad (5)$$

$$h_{T2}(y) = \begin{cases} 1-12y, & \varepsilon_T = 0 \\ 1-\dfrac{d(y,\alpha_T+\varepsilon_T)}{d(\alpha_T+\varepsilon_T,\alpha_T-\varepsilon_T)}, & \varepsilon_T \ne 0, y > \alpha_T+\varepsilon_T \\ 1, & \varepsilon_T \ne 0, \alpha_T-\varepsilon_T \le y \le \alpha_T+\varepsilon_T \\ 1-\dfrac{1.5d(y,\alpha_T-\varepsilon_T)}{d(\alpha_T+\varepsilon_T,\alpha_T-\varepsilon_T)}, & \varepsilon_T \ne 0, y < \alpha_T-\varepsilon_T \end{cases} \quad (6)$$

where $d(a,b)$ denotes Euclidean distance between a and b, y= $f_1(x)$ in Eq.(5), y= $f_2(x)$ in Eq.(6), α_T is ε_T standard scale of predicate P, and α_F is ε_F standard scale of predicate ╕P that is the inverse opposite of predicate P.

Definition 2. Ten predicates used for numeral character recognition are defined as: $A_i(x)$: the numeral a_i is in the image x, $0\leqslant i\leqslant 9$, where $x\in X$, and the object set X={segmented degraded images}, a_0~a_9 denote 10 numeral characters ('0'~'9') to be identified.

Definition 3. The truth degree function of X relative to $A_i(x)$, $0\leqslant i\leqslant 9$ defined in Def. 2, that is, $g_{nT\text{-}M\text{-}i}$: $X\rightarrow\boldsymbol{R}$ is defined as:

$$g_{nT-M-i}(x) = \left(\sum_{k=0}^{28} w_{i-k} h_{Ti-k}(m_k(x))\right) \Big/ \left(\sum_{k=0}^{28} w_{i-k}\right) \quad (7)$$

where $w_{i\text{-}0}$~$w_{i\text{-}28}$ denote the weights of features m_0~m_{28} (denoted as X_0~X_{28} in Secs. 2.4 and 2.5) for identifying whether the numeral character a_i is in the image x, $m_k(x)$ denotes the value of the feature m_k for the image x, $h_{Ti\text{-}k}(m_k(x))$ denotes the distance ratio of $m_k(x)$ relative to $A_i(x)$, and adopting h_{T1} or h_{T2} defined in Def. 1 to calculate $h_{Ti\text{-}k}$ depends on the type (i.e. 1 or 2) of m_k for identifying the numeral character a_i.

3.2. *Parameter setting*

It is obviously that the numeral 1 differs greatly from other numerals in the feature X_{27}(i.e. m_{27}), thus only this feature is chosen to identify the numeral 1. Features in the following set are used for identifying other numerals:

S_{subf} = {X_0~X_3, X_5~X_8, X_{11}, X_{14}, X_{16}, X_{17}, X_{20}, X_{23}, X_{26}, X_{28}} (8)

If a feature is chosen, then its weight in Eq. (7) is simply set to 1, otherwise 0. Thus, $w_{1\text{-}27}$ is set to 1, and $w_{1\text{-}28}$ as well as $w_{1\text{-}k}$ (k=0,1,2,…26) are set to 0; $w_{i\text{-}0}$~$w_{i\text{-}3}$, $w_{i\text{-}5}$~$w_{i\text{-}8}$, $w_{i\text{-}11}$, $w_{i\text{-}14}$, $w_{i\text{-}16}$, $w_{i\text{-}17}$, $w_{i\text{-}20}$, $w_{i\text{-}23}$, $w_{i\text{-}26}$ and $w_{i\text{-}28}$ (i=0,2,3,4,…9) are set to 1, and other $w_{i\text{-}k}$ are set to 0.

The type of m_{27} is 1, while types of other features are all 2. Four parameters in Eq.(5) must be set for calculating $h_{T1\text{-}27}(m_{27}(x))$. Two parameters in Eq.(6) must be set for computing each $h_{Ti\text{-}k}(m_k(x))$ where $i\neq1$ and $k\neq27$. Training samples (80 samples used in this paper) are used to determine parameters for the distance ratio functions as follows:

For each numeral ***i***, do {

Compute features of samples for numeral ***i*** according to Secs. 2.1~2.4;

For each feature m_j of this numeral, do {
Sort values of m_j for samples with the same numeral i in ascending order and obtain an ordered list;
Let $p0$, $p1$, $p2$ be 20th, 50th, 80th percentile of the list respectively, and $p3$ = max(0.12*$p1$, min($p2$ - $p1$, $p1$ - $p0$));
If the numeral is 1 and the feature is m_{27}, then set α_T, ε_T, α_F, ε_F for Eq.(5), to be $p1$, $p3$, $p1$*0.4, $p3$*0.4 respectively;
If the numeral is not 1 and the feature is in the set S_{subf} defined in Eq.(8), then set α_T, ε_T for Eq.(6), to be $p1$, $p3$ respectively;
}
}

3.3. *Fuzzy classifiers and basic recognition algorithm*

To discern which numeral is in an image, a total of 10 fuzzy classifiers are constructed according to Eq.(7) where parameters are set according to Sec. 3.2. Each classifier produces a truth degree that a certain numeral is in an image. Then, we can derive two most possible numerals in the image from ten truth degrees.

We design the following basic offline recognition algorithm using ten classifiers to identify degraded numeral characters given features of an image x:

1) Compute ten truth degrees $g_{nT\text{-}M\text{-}i}$ $(0 \leqslant i \leqslant 9)$ according to Eq.(7);

2) If $g_{nT\text{-}M\text{-}1}$<0.1, then save the value and set it to be $-\infty$;

3) Find the highest truth degree denoted as $g_{nT\text{-}M\text{-}k1}$, and the second highest one denoted as $g_{nT\text{-}M\text{-}k2}$;

4) Reset $g_{nT\text{-}M\text{-}1}$ to be the saved value if it is $-\infty$; return the likeliest numeral $k1$, the second likeliest numeral $k2$, $g_{nT\text{-}M\text{-}k1}$, and $g_{nT\text{-}M\text{-}k2}$.

3.4. *CBM-based recognition*

To improve the recognition accuracy, we studied further degraded images. We find that characters leaning a little to the left or to the right in an image are not unusual. So, to recognize the numeral in an image, we not only discern the original image, but also discern the left rotated image and the right-rotated image derived from the original image if needed. The likeliest numeral with the maximal value among three highest truth degrees for three kinds of images may be regarded as the numeral in the image.

We present the following rotation algorithm to rotate a monochrome image:

1) Assume a monochrome image as shown in Fig. 3 is to be turned left or right, point C(*midi*, *midj*) with *midi* = *height*/2 and *midj* = *width*/2 is the center pixel of the image, point A(i, j) is a pixel in the foreground, and point A'(x, y) is the point A after the image is rotated.

We use positive values to represent radians of left turn, while negative values for radians of right turn. Figure 3 shows that point A is turned left to A' with α radians.

2) For each foreground pixel A(i, j), do the following to determine its position (x, y) in the rotated image:

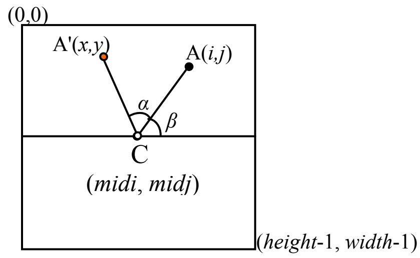


Fig. 3. Rotating an image

Set β= arcsine ((*midi*-*i*)/|AC|);
If (j<=*midj*), then
x = *midi* - |AC|*sin(β-α); y = *midj* - |AC|*cos(β-α);
Else
x = *midi*-|AC|*sin(β+α); y = *midj* + |AC|*cos(β+α);

The whole recognition procedure is described as follows:

1) Use training samples with several samples for each numeral to construct ten classifiers according to Eq.(7) where weights as well as parameters for the distance ratio functions are set as stated in Sec. 3.2.

2) Set LIMIT1=0.75, LIMIT2=0.75, LIMIT3=0.3;

3) For a degraded gray image x to be identified, do {
Compute the features of the image x according to Secs. 2.1~2.4;
Call the basic offline recognition algorithm in Sec. 3.3, and then set $d11$, $d12$, *truth*11 and *truth*12 to be the returned four values (**i.e. *k*1, *k*2, $g_{nT\text{-}M\text{-}k1}$ and $g_{nT\text{-}M\text{-}k2}$**) respectively;
If (*truth*11>=LIMIT1 or (*truth*11>=LIMIT3 and *truth*11-*truth*12 >= LIMIT2)), then
Set $d1$ = $d11$, and $d2$ = $d12$;
Else {

Compute the features of the left-rotated image of the image x according to Sec. 2.1, the rotation algorithm in Sec. 3.4 with α=10π/180, and Secs. 2.2~2.4;
Call the basic offline recognition algorithm, and then set d21, d22, *truth*21 and *truth*22 to be the returned four values respectively;
If (*truth*21>=LIMIT1 or (*truth*21>=LIMIT3 and *truth*21-*truth*22 >= LIMIT2)), then
    Set d1 = d21, and d2 = d22;
Else {
    Compute the features of the right-rotated image of the image x according to Sec. 2.1, the rotation algorithm with α=-10π/180, and Secs. 2.2~2.4;
    Call the basic offline recognition algorithm, and then set d31, d32, *truth*31 and *truth*32 to be the returned four values respectively;
    Set d1=di1, and d2=di2, where *truth*i1 is the max of *truth*11, *truth*21, and *truth*31;
  }
}
Assert d1 to be the likeliest numeral, and d2 to be the second likeliest numeral in the image x.
}

4. Experimental Results

We have implemented a numeral character recognition system according to the proposed method, and adopt 400 degraded gray images from Ref. 14 as the test dataset, where only a total of 80 relatively legible as well as upright images with 8 samples for each numeral are taken as training samples.

First we show recognition results of DNRBM for some images including two images in Fig. 1 in Table 1, where d11, d12, d21, d22, d31, d32, d1 and d2 refer to Sec. 3.4. It shows that the numeral in a degraded image can often be deduced just from recognition results for the original image; however, when there is no truth degree remarkable enough in ten truth degrees for the original image, it is necessary to rotate the image and discern further, that renders the recognition accuracy improved.

Next we show the process of locating the character in a degraded gray image by DNRBM in Fig. 4 and Fig.5. They show that both distinguishing foreground from background and wiping off mottles contribute to clearing blurs and converting a degraded gray image to a clear image; while rotating can make an image more upstanding that is beneficial to classification as shown in Table 1.

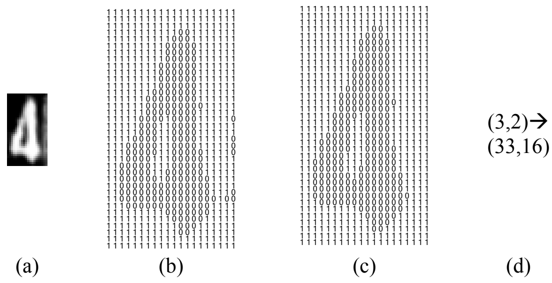


Fig. 4. Locating the character in a degraded gray image by DNRBM (a) original image (b) after distinguishing foreground from background (c) after wiping off mottles (d) the location of the foreground got after cutting margins

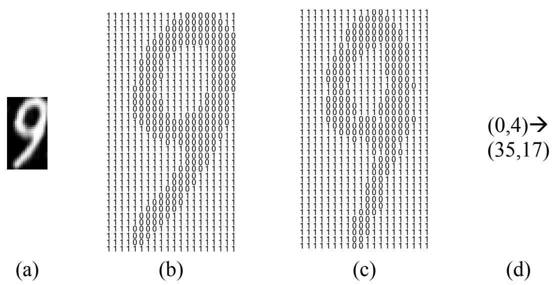


Fig. 5. Locating the character in a leaning gray image by DNRBM (a) original image (b) after distinguishing foreground from background (c) after turning left with 10 degrees (d) the location of the foreground got after cutting margins

Next, we compare the recognition accuracy using our method (DNRBM) and the template matching method (TMM)[10] implemented by Ref. 14 in Table 2. Experimental results show that DNRBM recognition rates for numerals 4,5,7,9 are notably higher than the TMM; when only the likeliest numerals (d1) considered as the recognition results, the average recognition rate of DNRBM reaches 93% with 11% higher than TMM. When considering both d1 and d2, a recognition error means that neither d1 nor d2 is the actual numeral, and the average recognition rate increases to 97%.

Experimental results also show that our method is a little worse in some cases. With regard to 40 test samples of numeral 8, we find 7 samples are misclassified as numeral 5. As to 40 test samples of numeral 3, 4 samples are misclassified as numeral 4, and 3 ones are misclassified as numeral 7. The misclassifications may mainly attribute to: (1) features not good enough for discriminating the numeral pairs "8-5", "3-4", "3-7", "5-6"; (2) rotating the character in an image without adapting well to the leaning degree; and (3) false distinguishing foreground from background.

Table 1. DNRBM recognition results for some images.

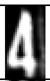
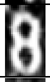
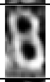
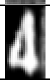
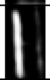
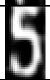
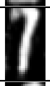
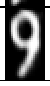
Images	Truth degrees of the image relative to $A_i(x)$, i=0,1,2,..,9		the likeliest numeral($d1$)	the 2[nd] likeliest numeral($d2$)
	-1.79 0.00 -2.46 -2.28 **0.63** -1.49 **-0.77** -2.01 -1.79 -2.51,	$d11$=4, $d12$=6	4	6
	**0.68** -0.06 -1.11 -2.52 -0.36 0.46 -0.32 -1.37 **0.83** -0.58,	$d11$=8, $d12$=0	8	0
	**0.05** -0.26 -0.72 -2.32 -0.20 0.02 -0.10 -1.24 **0.18** -1.19, -0.74 -0.29 -1.26 -2.49 -0.64 **-0.38** **0.18** -1.45 -0.69 -1.44, -0.83 -0.29 -1.17 -2.61 **-0.03** -1.07 **-0.23** -0.98 -0.77 -1.44,	$d11$=8, $d12$=0 $d21$=6, $d22$=5 $d31$=4, $d32$=6	8	0
	-1.64 -0.10 -2.09 -2.34 **0.68** -1.48 **-0.65** -1.84 -1.64 -2.42,	$d11$=4, $d12$=6	4	6
	-1.57 -0.14 -0.66 **0.02** **-0.19** -1.95 -1.15 -1.05 -1.63 -1.90, -2.08 0.00 -1.13 **0.34** -1.08 -1.55 **-0.60** -1.18 -2.08 -2.06,	$d11$=3, $d12$=4 $d21$=3, $d22$=6	3	6
	-1.47 -0.30 -2.50 -3.37 **0.06** -1.32 **-0.42** -2.08 -1.46 -2.14, -3.05 -0.18 -3.29 -4.78 **-1.19** -3.11 **-0.30** -2.46 -3.18 -3.61, -1.73 -0.44 -2.25 -3.82 **0.10** -2.12 **-1.24** -1.64 -1.90 -2.18,	$d11$=4, $d12$=6 $d21$=6, $d22$=4 $d31$=4, $d32$=6	4	6
	-4.49 **1.00** -2.45 -2.43 -3.29 -4.56 -2.76 **-1.21** -4.90 -3.83,	$d11$=1, $d12$=7	1	7
	-0.36 0.00 -1.08 -1.30 -0.97 **0.77** -0.52 -1.60 **-0.30** -0.77,	$d11$=5, $d12$=8	5	8
	-0.84 -0.14 -1.71 -2.19 -1.51 **-0.01** **-0.76** -2.32 -0.94 -1.19, -1.48 -0.25 -1.80 -2.67 -1.67 **-0.56** **-1.03** -1.97 -1.52 -1.12, -1.08 0.00 -1.05 -1.05 -0.99 **-0.61** **-0.11** -1.48 -0.94 -1.28,	$d11$=5, $d12$=6 $d21$=5, $d22$=6 $d31$=6, $d32$=5	5	6
	-0.77 -0.14 **0.50** -0.81 -1.03 -1.65 -1.91 **-0.17** -0.88 -1.26, -3.22 -0.21 **-1.86** -2.06 -2.80 -2.73 -2.20 **-1.17** -3.43 -2.86, -2.42 -0.21 **-0.66** -1.57 -0.68 -3.35 -2.53 **-0.51** -2.60 -2.46,	$d11$=2, $d12$=7 $d21$=7, $d22$=2 $d31$=7, $d32$=2	2	7
	-2.92 0.00 **-1.39** -1.87 -2.49 -3.91 -3.80 **0.42** -3.27 -2.04,	$d11$=7, $d12$=2	7	2
	**0.02** -0.03 -0.92 -3.11 -0.71 -1.08 -1.70 -0.43 **0.03** -0.31, -0.62 0.00 -1.27 -2.97 -0.94 -1.27 -1.62 **-0.19** -0.76 **0.08**, -1.68 -0.28 -1.60 -2.05 **-0.57** -2.68 -2.31 **-0.73** -1.80 -1.66,	$d11$=8, $d12$=0 $d21$=9, $d22$=7 $d31$=4, $d32$=7	9	7
	**-0.34** 0.00 -1.20 -2.84 -1.00 -0.76 -1.84 -0.56 -0.39 **0.89**,	$d11$=9, $d12$=0	9	0

Table 2. Comparisons of recognition accuracy

Numeral	Total	Recognition errors		
		TMM	Our method (DNRBM)$d1$ as the result	Our method (DNRBM)both $d1$ and $d2$ as the results
0	40	0	0	0
1	40	2	3	2
2	41	6	2	1
3	40	5	7	7
4	40	17	0	0
5	40	11	3	0
6	39	2	2	0
7	40	13	3	1
8	40	6	8	1
9	40	10	0	0
Average recognition rate		82%	93%	97%

Finally, we have implemented k-NN classifiers based recognition method for comparison with DNRBM since k-NN classifiers usually achieve high performance. The k-NN classifiers take X_0~X_{26} or X_0~X_{28} as features that are calculated according to Secs. 2.1, 2.3 and 2.4, use the same training and test sets as DNRBM, and adopt Euclidean distance metric and weighted or simple voting scheme. The weights for the nearest neighbor and the second nearest one for 2-NN are 0.6, 0.5 respectively, 0.5, 0.3, 0.25 for 3-NN, 0.5, 0.4, 0.3, 0.25 for 4-NN, and 0.5, 0.4, 0.3, 0.25, 0.22 for 5-NN. Table 3 shows the recognition accuracy for k-NN classifiers with different k, features and the voting scheme. In comparison with the k-NN classifiers based recognition method, DNRBM reveals better recognition accuracy.

Table 3. Recognition accuracy for the k-NN classifiers based recognition method

k	Weighted voting		Simple voting	
	Features (X_0~X_{26})	Features (X_0~X_{28})	Features (X_0~X_{26})	Features (X_0~X_{28})
1	0.8950	0.9150	0.8950	0.9150
2	0.8950	0.9150	0.8850	0.8925
3	0.8825	0.8975	0.8850	0.8875
4	0.8875	0.8875	0.8825	0.8625
5	0.8775	0.8925	0.8575	0.8750

5. Conclusions

The contribution of this paper is fourfold. First, an effective approach is proposed to locate the character in a degraded gray image. Secondly, this work has proved that using logistic regression to obtain a reduction in the number of features is effective. Thirdly, this work has used both statistic and structural features, and experimental results highlight that integrating various features of objects for recognition is usually a good idea. Finally, this work indicates again that the measure of *n*-dimensional medium truth degree is suitable for multi-class classification for its scalability[15], and moreover, classification based on MMTD performs well when only a small number of samples available for training. We plan to work on the proposed method DNRBM further to promote the recognition accuracy by designing more simple and useful features, as well as by designing more intelligent algorithms to distinguish foreground from background, to rotate images more adaptively, to wipe off mottles, and to refine results of fuzzy classifiers to make a final decision.

Acknowledgements

The authors wish to thank the reviewers for their constructive comments. This work was partially supported by the National Natural Science Foundation of China under Grant No.61170322 and Grant No. 70871061, by the Open Fund of the State Key Laboratory of Software Development Environment under Grant No. SKLSDE-2011KF-04 (Beihang University), and by the project funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions (Information and Communication, yx002001).

References

1. M. T. Parvez and S. A. Mahmoud, Offline arabic handwritten text recognition: A survey, *ACM Computing Surveys*. **45**(2) (2013), Article 23, 35 pages.
2. Adnan Amin, Off-line Arabic character recognition: the state of the art, *Pattern Recognition*. **31**(5) (1998) 517–530.
3. L. M. Lorigo and V. Govindaraju, Offline Arabic handwriting recognition: a survey, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **28**(5) (2006) 712–724.
4. L. Likforman-Sulem and M. Sigelle, Recognition of degraded characters using dynamic Bayesian networks, *Pattern Recognition*. **41**(10) (2008) 3092–3103.
5. V. Vaidya, V. Deshpande, D. Gada and B. Shirole, Statistical approach to feature extraction for numeral recognition from degraded documents, in *Proceedings of the Fifth International Conference on Document Analysis and Recognition* (Bangalore, 1999), pp. 273–276.
6. C. Liu, K. Nakashima, H. Sako and H. Fujisawa, Handwritten digit recognition: benchmarking of state-of-the-art techniques, *Pattern Recognition*. **36**(10) (2003) 2271–2285.
7. S. Abdelazeem, Comparing arabic and latin handwritten digits recognition problems, *World Academy of Science, Engineering and Technology* 30. (2009) 446–450.
8. S. Mozaffari, K. Faez and M. Ziaratban, Structural decomposition and statistical description of Farsi/Arabic handwritten numeric characters, in *Proceedings of IEEE Eighth International Conference on Document Analysis and Recognition* (Seoul, South Korea, 2005), pp. 237–241.
9. N. Arica and F. T. Yarman-Vural. An overview of character recognition focused on off-line handwriting, *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*. **31**(2) (2001) 216–233.
10. Ivind Due Trier, Anil K. Jain and Torfinn Taxt, Feature extraction methods for character recognition--a survey, *Pattern Recognition*. **29**(4) (1996) 641–662.
11. L. Hong, X. Xiao, W. Zhu, Measure of medium truth scale and its application (I), *Chinese Journal of Computers*, **29**(12) (2006) 2186−2193 (in Chinese).
12. X. Xiao, W. Zhu, A system of medium axiomatic set theory, *Scientia Sinica (Series A)*, **XXXI**(11) (1988) 1320–1335.
13. S. Ma and J. Huang, Penalized feature selection and classification in bioinformatics, *Briefings in Bioinformatics*. **9**(5) (2008) 392–403.
14. http://www.verysource.com/handwrittenfiguresid-16142.html (2011).
15. W. CHENG, J. GONG, W. DING and Z. SUN, Application type identification of Internet flows based on Medium Mathematics, *The Journal of China Universities of Posts and Telecommunications*. **17**(6) (2010) 72–79,88.