Research on Influence of Image Preprocessing on Handwritten Number Recognition Accuracy

Tieming Chen(✉), Guangyuan Fu, Hongqiao Wang, and Yuan Li

Xi'an Research Institute of High-Technology, Xi'an 710025, Shaanxi, China
chentieming1995@163.com

Abstract. In the process of handwritten number recognition, image pretreatment is a key step that has a great influence on the recognition accuracy. By unifying the standard, handwritten digital images are normalized, which can improve the adaptability of handwritten digital recognition algorithms to different writing habits. This article mainly considers the four characteristics of the angle, position, size and strength when writing characters, and how these factors influence four classical handwritten recognition algorithms. According to the four characteristics, tilt correction, offset correction, size normalization and thinning preprocessing were performed one by one to observe the changes of recognition accuracy in four classical algorithms. Through experiments, it is found that the recognition accuracy of the original data set and the scrambling data set are both greatly improved after preprocessing operation. In conclusion, it is resultful to increase the recognition accuracy by image preprocessing in handwritten digital recognition.

Keywords: Image preprocessing · Handwritten number recognition · CNN · Pretreatment

1 Introduction

As a branch of Optical Character Recognition (OCR), handwritten digital recognition has been a very hot research topic in the field of image recognition [1–3]. The essence of recognition is the classification of pictures, that is, the algorithm matches the largest similarity number as the final classification result. Although there are only ten numbers, and the strokes are relatively simple, it is extremely difficult to identify handwritten digits due to the different handwriting habits. To solve this problem, it is critical to find a universal processing method to improve the accuracy of recognition.

Up to now, there have been plenty of studies on image preprocessing, most of which are brief summary methods [4–6]. This article explains in detail how to perform image preprocessing in handwritten digit recognition and also simplifies the complexity and difficulty of method, when meeting the accuracy requirement. In addition, this paper proposes a preprocessing solution especially for four kinds of handwriting habits from different perspectives of angles, positions, sizes, and strengths when people writing characters. In order to verify the influence of preprocessing from various aspects, this

Q. Liu et al. (Eds.): CENet 2018, AISC 905, pp. 253–260, 2020.
https://doi.org/10.1007/978-3-030-14680-1_28

paper uses four classical algorithms. By comparing the experimental results with or without preprocessing, the influence of image preprocessing can be seen clearly.

2 Image Preprocessing

2.1 Character Segmentation

The character segmentation is the basis of image pretreatment, which identifies the area needed to be processed. The image is first grayed and binarized. The binarization process makes every pixel pure black (pixel value is 0) or pure white (pixel value is 255). These grayscale images can help speed up calculations and improve recognition accuracy.

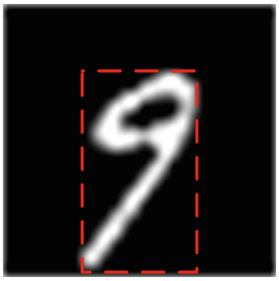

Fig. 1. The character zone.

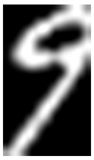

Fig. 2. The character segmentation image.

The core of the character segmentation algorithm is to find the coordinates of the four vertices on the character border. The process is as follows:

First, input the image that has been grayed and binarized;

In the second step, scan the input image progressively, and compare pixel value of each point with the size of 255 (when the pixel value is 255, it represents white, that is, the foreground color of the handwritten digital image). Then record the coordinates of points where the pixel value is equal to 255.

Third, repeat the second operation until all the pixels are traversed. Then the minimum horizontal (vertical) coordinate of all recording points is left margin (bottom), which is denoted by $x_{min}(y_{min})$. Similarly, the maximum horizontal (vertical) coordinate founded is the right margin (top), which is denoted by $x_{max}(y_{max})$.

Forth, the segmented character area can be shown in Figs. 1 and 2, which can be described as $S = \{(a, b)|x_{min} \ll x \ll x_{max}, y_{min} \ll y \ll y_{max}\}$.

2.2 Tilt Correction

When a person writes numbers with different habits, the characters often have a certain angle of tilt. Therefore, the angle of characters need to be corrected. At the same time, the tilt-corrected character image also facilitates the segmentation of the character and improves the recognition accuracy.

In general, the character image after character segmentation should have similar height of left and right pixels. If the average height of pixels on both sides has a large difference, it indicates that the image is tilted and needs to be adjusted. So adjust the image by using the average height of the left and right white pixels of the image, the main process of tilt correction is shown in Fig. 3.

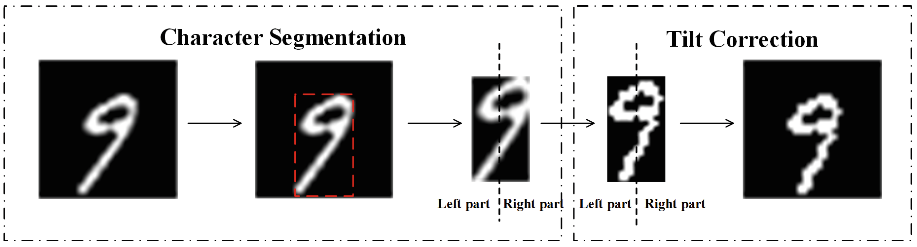

Fig. 3. The tilt correction after character segementation.

First, calculate the average weighted height of the white pixels in the left part and right part of image. The rule for the weight setting is that the weight is higher when the pixel is closer to two edges. Then calculate the slope of inclination and reorganize the image:

$$S = \frac{H_l - H_r}{W/2} \tag{2.1}$$

In the above equation, S denotes the slope of inclination, H_l and H_r represent the average weighted height in the left and right half image respectively, W represents the width of a character image. Then, the coordinates of corrected point can be calculated as:

$$\dot{x}_i = \lfloor x_i - (y_i - W/2) \times S \rfloor, \dot{y}_i = y_i \tag{2.2}$$

where $(\dot{x}_i, \dot{y}_i)$ denotes the coordinate of the point after the inclination correction of the point (x_i, y_i), and $\lfloor x \rfloor$ indicates rounding down to x.

2.3 Offset Correction

In most cases of writing numbers, the center of character will deviate from the center of the image, which can easily affect the recognition accuracy. And the center-of-gravity normalization method is usually used. First, calculate the center of gravity A in the image, which should have coincided with the center of the image B. Point A and point B can make up an offset vector $\boldsymbol{AB}$. Later, the handwritten digit offset can be corrected by moving the red frame in Fig. 4 in accordance with the direction and distance of the offset vector. The formula for calculating the center of gravity (x_h, y_h) of the image is:

$$\begin{cases} x_h = \frac{\sum_{i=1}^{n} x_i}{n} \\ y_h = \frac{\sum_{j=1}^{n} y_j}{n} \end{cases} \tag{2.3}$$

where n is the number of white points, $\sum_{i=1}^{n} x_i$ represents the sum of the abscissas of white points, and $\sum_{j=1}^{n} y_j$ represents the sum of the vertical ordinates of white points.

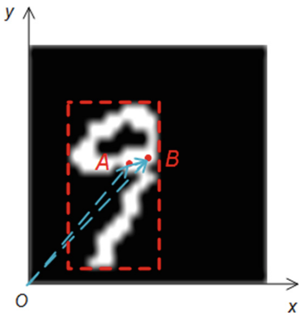

Fig. 4. The center of gravity and center of the image.

As shown in Fig. 4, the center of gravity of image is $A(x_h, y_h)$. Construct a direction vector AB $= (x_H - x_h, y_H - y_h)$ with its center $B(x_H, y_H)$.

$$\begin{cases} \dot{x} = x + (x_H - x_h) \\ \dot{y} = y + (y_H - y_h) \end{cases} \tag{2.4}$$

In the above equation, $\dot{\mathrm{P}}(\dot{x}, \dot{y})$ is a new coordinate point obtained after offset correction from point $\mathrm{P}(x, y)$, and $x_{min} \ll x \ll x_{max}, y_{min} \ll y \ll y_{max}$. After moving all the white pixels in the red box by the vector AB, the center of gravity can finally coincide with the center of image, and the corrected image is shown in Fig. 5.

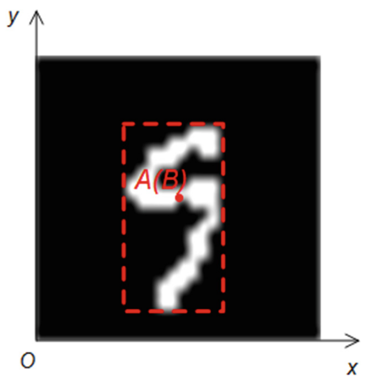

Fig. 5. The center of gravity and center of the image coincide.

2.4 Size Normalization

The purpose of size normalization is to make the length and width ratio of the segmented character images basically similar, so as to eliminate the interference caused by the inconsistent handwritten digits. The advantage is to improve the accuracy of image recognition after unified size. Common image size normalization methods include linear normalization and nonlinear normalization. The nonlinear normalization method is based on the linear normalization method and it adds nonlinear change factors which can better solve the problem of image deformation after scaling. In the actual writing process, the handwritten digit size difference will be small, i.e., the influence of the deformation of the image before and after the normalization process will not cause too much influence on the recognition accuracy.

And compared with the nonlinear normalization method, the linear normalization method has a simpler method, a smaller amount of calculation, so use the linear normalization method which can also meet the image scaling requirements.

The linear normalization method is an image scaling method that linearly zooms in or zooms out the original image to a predetermined size in a certain proportion. The linear normalization formula is given now:

$$\begin{cases} m = \frac{i \times M}{I} \\ n = \frac{j \times N}{J} \end{cases} \tag{2.5}$$

The point (m, n) indicates the normalized coordinate point corresponding to original point (i,j).

Taking into account the characteristics of the digital shape, it is specified that the image of number one after segmentation is normalized to 20×6, and the rest is normalized to 20×14. The algorithm is to obtain the height and width of the segmented character image firstly. Then the height and width are compared to obtain the transform coefficient. Later, the points in the new image are mapped to the original image according to the interpolation method.

2.5 Image Thinning

Different writing strength leads to different thicknesses of digital strokes, which will have a great impact on extracting feature values in the process of digital recognition. In order to enhance the adaptability of digital recognition, the process of image thinning is necessary.

A good image thinning algorithm has the following requirements:

1. The skeleton image must maintain the connectivity of the original image.
2. The skeleton image should be the centerline of the original image as much as possible.
3. The thinning skeleton should be thin for a pixel-wide line image as much as possible.
4. The algorithm should use as few iterations as possible.

There are already many kinds of image thinning algorithms such as Classical Image Thinning Algorithm, Pavlidis Asynchronous Image Thinning Algorithm, Deutseh Algorithm, Zhang Fast Parallel Image Thinning Algorithm and so on. Among them, Zhang's fast parallel image thinning algorithm is a practical thinning method. It is an 8-adjacent and parallel image thinning algorithm, which has the advantage of high speed and ability to maintain the connectivity of curves. Due to the disadvantage of missing local information and the presence of redundant pixels after thinning, this paper uses the improved Zhang fast parallel image thinning algorithm to thin handwritten characters [7].

Because of ink brushing, thick strokes, or overly random writing, handwritten numbers are prone to black blocks, especially numbers 0, 6, 8, and 9, as shown in Fig. 6. And Fig. 7 is the image after binarization, size normalization, tilt, and offset correction. If use the algorithm proposed by Zhang directly, as shown in Fig. 8, discrete and irregular lines appear in image, which losts the original structure and information of the character. The use of the improved algorithm proposed by Zhang Cuifang and others for such problems can effectively maintain the shape of the characters after thinning, as shown in Fig. 9.

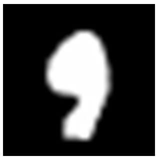
Fig. 6. The original image

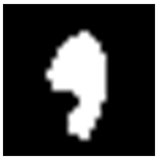
Fig. 7. The binary image

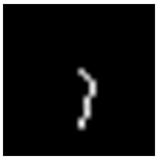
Fig. 8. The thinning corrected image using Zhang's algorithm

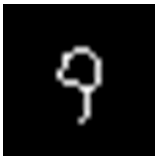
Fig. 9. The thinning corrected image using improved Zhang's algorithm

3 Result and Conclusion

The data used in the experiment comes from the MNIST Handwritten Numeric Character Database, which contains 70,000 handwritten digital samples from 0 to 9 divided into 60,000 training data sets and 10,000 test data sets. The shape of all pictures are 28 × 28. And the experimental environment is: 64-bit Windows 10 operating system, programmed by Python, the four classical algorithms used are k-Nearest-Neighbor (KNN), Support Vector Machine (SVM), Back Propagation Neural Network (BP) and Convolutional Neural Network (CNN). The experimental steps are as follows:

In the first step, the MNIST raw data set is grayed and binarized, and four classical algorithms are used for training and testing. Then, the original data is preprocessed, with parameters still unchanged, and four algorithms are reused. Identify and obtain experimental results as shown in Table 1.

In the second step, in order to explore the influence of different writing habits on the recognition accuracy of handwritten digits, the MNIST data set is scrambled. The method is that 25% of samples are randomly selected from the training set and the test set, then perform position offset, angular offset, size scaling, and dilation erosion operations respectively, which simulates the writing habits of different people. Keep the same algorithm parameters, train and test new data sets, and the result is shown in Table 2.

The third step is to preprocess the scrambled data set obtained in the second step, that is, perform the tilt correction, offset correction, size normalization and thin images operations. Then use the four classical algorithms and the result is shown in Table 2.

The fourth step is to do experiments for multiple times, compare, analyze data, and make a conclusion.

It can be seen from Table 1 that the four algorithms all have a good representation on MNIST original data set. After the data set is preprocessed, only the recognition rate of KNN algorithm has a slight decrease. The recognition rate of the other three algorithms has a little increased, and CNN algorithm can improve even up to 99.31%. Through experiments, the recognition of number 1 in four algorithms after preprocessing is generally improved a lot. What's more, the recognition rate of number 7 is also improved which has a similar shape and simple structure with number 1.

Observe the result in Table 2 and it is found that compared with the original data set, the recognition rates of the first three algorithms with scrambled data set are all reduced, while the recognition rate of the CNN algorithm has been improved, still lower than the original data set after preprocessing. The possible reason is that the diversity of the sample is increased after scrambling, and from the side it also reflects the strong learning ability of the deep learning algorithm.

Table 1. Handwritten digit recognition accuracy results before and after preprocessing on raw data set.

Algorithm	KNN		SVM		BP		CNN	
Condition	Before	After	Before	After	Before	After	Before	After
Average accuracy rate (%)	96.85	96.52	94.39	95.54	97.37	97.47	97.56	99.31

Table 2. Handwritten digit recognition accuracy results before and after preprocessing on scrambled data set.

Algorithm	KNN		SVM		BP		CNN	
Condition	Before	After	Before	After	Before	After	Before	After
Average accuracy rate (%)	88.53	91.82	89.56	91.22	95.77	97.43	98.61	99.43

The above data shows that the preprocessing of handwritten digital images is very necessary for the traditional machine learning algorithms like KNN and SVM algorithm. Meanwhile, the neural network algorithm has higher adaptability to the recognition of handwritten digits with different characteristics. Because of the strong self-adaptive and self-learning ability of deep learning, the scrambled data set to some extent increases the diversity of input samples and further improves the recognition accuracy of CNN. In conclusion, whether it is machine learning or deep learning method, the preprocessing of handwritten images can achieve improved recognition accuracy generally.

References

1. Zhang, M., Yu, Z.Q., Yao, S.W.: Image pretreatment research in recognition of handwritten numerals. Microcomput. Inf. **22**(16), 256–258 (2006). (in Chinese)
2. Chen, H., Guo, H., Liu, D.Q., Zhang, J.Q.: Handwriting digital recognition system based on tensorflow. Inf. Commun. **3** (2018). (in Chinese)
3. Zhang, Y.S.: License plate recognition key technology related algorithms research and implementation. North Minzu University (2017). (in Chinese)
4. Zhang, C.F., Yang, G.W.: Research on image pretreatment technique in recognition of handwritten number. Comput. Sci. Appl. **6**(6), 329–332 (2016). (in Chinese)
5. Huang, Q.Q.: Research on handwritten numeral recognition system based on BP neural network. Central China Normal University (2009). (in Chinese)
6. Zhang, S.H.: Study and realization of algorithms for chinese characters image's preprocessing. Microcomput. Dev. **13**(4), 53–55 (2003). (in Chinese)
7. Zhang, C.F., Yang, G.W., Yue, M.M.: Improving of Zhang parallel thinning algorithm. Inf. Technol. Informatiz. **6**, 69–71 (2016). (in Chinese)