“0” and “O” Recognition Based on Deep Learning

Wuyi Xiao, Junxian Ma, Huazhu Liu and Chunping Liao

ABSTRACT

The traditional algorithm has achieved good recognition for the recognition of most characters, but the recognition rate of the number “0” and the letter “O” is only 80-90%, which is difficult to meet the actual needs of the industry. For the recognition of these two characters, a recognition method based on deep learning is proposed. First, image preprocessing is performed on the characters, and then the sample data is manually labeled, and 40,000 training samples and 10,000 test sample images are obtained by the data samples enhancement. The results show that the CNN network can achieve more than 99% recognition rate, the training sample time is about 5 minutes, and 1000 images can be recognized in 1 second. Both the recognition speed and the recognition effect can meet the actual needs of the industry.

KEYWORDS

INTRODUCTION

Prior to the advent of deep learning, the mainstream algorithms were the connected region-or HOG-based detection methods that were based on the traditional handcraft features[1][2]. For example, the candidates for characters were derived through the maximally stable extremal regions (MSER), which were then regarded as the vertex of the graph.

At this time, the search for text lines could be considered as a process of clustering, since the texts from the same lines were usually identical in direction, color, font, and shape.

T. Siriteerakul et al.[3] empirically studied the classification of ThaiEnglish characters by using SVM [4] classifier with HOG features. Despite large numbers of characters of different classes, their method performed adequately well under some identifiable defects. The 2D graphic-based contour projection and morphological processing proposed by Chhaya Patel et al.[5] showed an accuracy of up to 89.24% for segmentation of handwritten text words. Chhaya Patel et al.[6]
proposed a Euclidean transformation-based region recognition method, whose accuracy for region recognition of handwritten text words reached 96.99%. S. Mandal et al.[7] put forward a frequent counting-based method utilizing the first, second derivatives and the slope and baseline characteristics, which showed an accuracy of 95.1% for handwritten character recognition. The accuracy rate of a SVM-based approach proposed by H. Nakkach et al.[8] was 92.43% for the identification and classification of Arabic characters. The deep learning-based convolutional neural network put forward by Shuye Zhang et al.[9] exhibited an accuracy of 98.44% for the recognition classification of similar Chinese characters.

The first step of conventional methods was image preprocessing, where the original image samples were processed to obtain the characters needing recognition, so that the subsequent feature extraction and learning could be performed. The primary purpose of this step was to remove redundant information from the sample images to better facilitate the subsequent processing. This process usually includes: grayscaling (for color sample images), denoising[10], binarization[11], character segmentation, and normalization. After binarization, the sample images only have two gray values of 0 and 255; among which 0 is the image background and 255 is the character under recognition. Denoising is vitally important at this stage, where the quality of noise reduction algorithm has a great influence on the feature extraction. Character segmentation divides the characters in an image into single characters that need recognition. Often, an additional slant correction will be required if the characters are slanted. In normalization operation, single character images are resized to the same size. A unified algorithm can be applied only under the same specification. Feature extraction (via various methods, e.g. PCA[12], or SIFT[13]): Features are key information used to identify characters, and each individual character can be distinguished from others by features. The SIFT-based off-line handwritten Chinese character recognition function proposed by Zhiyi Zhang et al.[14] could achieve a recognition rate of over 95%, which though failed to meet the expected level for hardly identifiable features. The classifier design, training, and recognition constitute the final step in character recognition. Classifiers[15] and[16] are used to perform recognition. For a character image, features are extracted as input to the classifiers, which then identify and classify these features.

Most of the industrial optical character recognition (OCR) techniques performed recognition detection based on traditional methods before the introduction of deep learning AlexNet[17]. However, the classification accuracy for the hard-to-distinguish “0” and “O” characters with the traditional recognition methods was undesirable after feature extraction. Therefore, deep learning CNN[18] [19] [20] [21] network model was used for their feature extraction, followed by training and test recognition with classifier, which improved the accuracy considerably (99% or higher).

RELATED WORK

Character recognition system consists of three modules. The first module is the image preprocessing stage. The preprocessing of images requires grayscaling, binarization, denoising, rectification, horizontal segmentation, and vertical segmen-
tation of original datasets to obtain the training samples. Furthermore, the training samples are labeled, and a dataset is created and denoted as tfrecord. The second module is the CNN network, which extracts feature vectors from each training sample image. The third module is a set of SoftMax[22].

ACQUISITION AND READING OF SAMPLE SETS

The core problem of identifying 0-O by deep learning training can be summarized, firstly, as about the sample quantity: the industrially acquired original images are 500 in number, which is rather small. For the training identification of small sample sets, an increase of samples is necessary. In this study, the dataset is increased by data enhancement[6]. For example, flipping, random trimming, color dithering, translation transformation, scale transformation, contrast transformation, noise perturbation, and rotation transformation/reflection transformation of images each adds 10 sample datasets to obtain 50,000 sample images in total. The second core problem faced by the training is the data reading mode. The length of training is decided largely by the way the data is read during training. Three methods of data reading are compared under the same sample training conditions. The first method is to read each training sample image, where the placeholder reads the data in the memory, and then feeds them to the holder variable via feed dict for value passing. The second method is to read the data in the hard disk by using queue. The third method is to prepare the data into tfrecord datasets before reading them using queue. In the GPU1080ti-based condition[23], training 10,000 steps with 5,000 training samples by method one took more than 5 hours, while with method three, only 250 seconds were consumed. The use of method three for data reading can address the time wastage caused by the training duration in the course of learning or engineering training. Table I lists the comparison of training durations:

<table>
<thead>
<tr>
<th>Methods</th>
<th>Method 1</th>
<th>Method 2</th>
<th>Method 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training time (secs)</td>
<td>19080</td>
<td>954</td>
<td>250</td>
</tr>
</tbody>
</table>

IMAGE PREPROCESSING

As shown in Figure 1, the original datasets collected are quite ideal, which can thus be binarized directly. The number “0” and letter “O” needing identification are vertically segmented from the original images, and then the segmented character samples are further segmented horizontally to remove the excess background part. This process adopts a pixel distribution based on sample images to segment each binarized original sample image. The segmentation method is to scan the pixels at each column and row of images, and set a flag bit when variation of pixel value (gray value changes from 0 to 255) is detected, then continue scanning. Another flag bit is set when variation of pixel value (gray value changes from 255 to 0) is scanned. Then, the area between the two flag bits is cut out to get the desired characters, as shown in
Figure 2. For the cut single characters, we extract the “0” and “O” we need, thereby obtaining the desired training dataset samples. This is the end of character segmentation part. After character extraction, the extracted character matrix is normalized into 64*64 matrix. Next, the segmented 0-O sample images are labeled and made into tfrecord datasets, as shown in Figure 3.

Figure 1. Original single images (500 in number).

Figure 2. Segmentation of single images in Figure 1.

Figure 3. Labeling of 0-O two types of data.

Data reading is performed on the obtained datasets by queue method. The reading thread continuously reads the images in the file system into a memory queue, while another thread is responsible for calculation. Data are read directly from the memory queue when required by the calculation. A queue is not just a data structure, which can also be an important mechanism for calculating the tensor value in one step. For example, multiple threads are able to simultaneously write elements to a queue or read elements in a queue. This can solve the idle GPU problem attributed to IO, thereby generating batches, which are then input into the CNN network for training.

DEEP LEARNING ALGORITHM

In recent years, deep learning has shown excellent performance in many fields of vision, voice and medical care. Among diverse deep neural networks, the best applicable one is convolutional neural network (CNN). It consists of three main layers: a convolutional layer, a pooling layer, and a fully-connected layer. The convolutional layer is used to learn the feature representation of input data. The pooling layer reduces the eigenvectors output by the convolutional layer, while improving the results (making the structure less prone to overfitting). The fully-connected layer
combines all local features into global features, which are used for calculating the scores for each of the last classes. Activation function introduces nonlinearity into the CNN.

Taking the LeNet-5 network[24] as an example, as shown in Figure. 4, the first hidden layer is convolved, which comprises six feature maps. Each feature map consists of 28*28 neurons, and each neuron specifies a 5*5 acceptance domain. The second hidden layer implements subsampling and local averaging, which also comprises 6 feature maps, except that each feature map consists of 14*14 neurons. The third hidden layer, which performs second convolution, comprises 16 feature maps, each consisting of 10*10 neurons. The fourth hidden layer, which is responsible for the second subsampling and local averaging calculation, comprises 16 feature maps, but each consisting of 5*5 neurons. The fifth hidden layer implements the last stage of convolution, which consists of 120 neurons, with each specifying a 5*5 acceptance domain. Finally, the fully-connected layer obtains the output class vectors.

![Figure 4. Structure schematic of LeNet-5 network.](image)

**EXPERIMENTS**

The CNN part adopts a LeNet5 network structure, in view of its relatively moderate training duration and number of training network layers.

**DESCRIPTION OF VARIOUS LENET-5 NETWORK PARTS**

- **Input:** 64*64 = 4,096 0-O character images, equivalent to 4,096 neurons. These character images contain the number 0 and the letter O, which equal to two categories of images.
- **C1 layer:** 16 feature convolution kernels are selected, whose size is 5*5, so that we can get 16 feature maps each with a size of 64-5+1=60. That is, the number of neurons is 16*60*60= 57600.
- **S2 layer:** Downsampling layer, which performs down-sampling by maximum pooling. The pooling filter size f is selected as (2, 2), whereas the stride is 2. This way, we can get 16 images each with an output size of 30*30.
- **C3 layer:** Convolutional layer, the size of convolution kernels remains 5*5, according to which we can derive a new image size of 30-5+1=26. Since 16 convolution kernels are adopted here, 16 images sized 26 * 26 are output finally.
- **S4 layer:** Downsampling layer, which performs maximum pooling on the 16 images sized 26*26 in C3. The pooling filter size f is selected as (2, 2), whereas the stride is 2. Accordingly, S4 layer has 16 images each with a size of 13*13. At this point, the number of neurons is reduced to: 16 * 13 * 13 = 2704.
- C5 layer: The output of S4 layer is tiled into a 2704 one-dimensional vector. Then, the next layer is constructed with these 2,704 neurons. The C5 layer contains 120 neurons. The 2,704 neurons in the S4 layer are connected to each neuron in the C5 layer (120 in total) to constitute the fully-connected layer, which can be regarded as a standard neural network layer.
- F6 layer: Another fully-connected layer is added to the C5 layer’s 120 neurons (the F6 layer contains 128 neurons).
- Output: Finally, the F6 layer’s 128 neurons are fed into a SoftMax function to derive a tensor with an output length of 2. The position 1 in the tensor represents the category.

**SOFTMAX**

Softmax regression model is a generalization of the logistic regression model concerning the multi-classification problem. For the multi-classification problem, the number of classes to be classified is greater than 2, and the classes are mutually exclusive. After the completion of convolutional, pooling, and fully-connected layers and data conversion to sample label space, binary classification of input images is needed (that is, we hope to calculate the conditional probability of each class conditioned by this image, a prior probability distribution). A column vector of length 2 can be derived via the fully-connected layer, which is a linear predictor generated by the network. Softmax is precisely a common function in machine learning for converting linear predictors into output probabilities, which is used to output the conditional probabilities for each class.

**ANALYSIS OF EXPERIMENTAL RESULTS**

The experiments reveal a recognition rate of 99.83% for testing the number “0” alone; and a recognition rate of 99.71% for testing the letter “O” alone. The experimental results are shown in Figure 5, respectively. TABLE II lists the comparison of recognition rate and time. The recognition of 10,000 long images takes only 3.86 seconds, with a recognition rate of 99.09%. Compared to the references [5-9], not only the recognition rate is improved greatly, but the recognition duration is also shorter, which can fully meet the online recognition requirements.

![Figure 5. Testing of number 0 alone(Probability 99.83%) and Testing of letter O alone (Probability 99.71%).](image-url)
TABLE II. TABLE TYPE STYLES.

<table>
<thead>
<tr>
<th>No. of test sample</th>
<th>No. of correct classification</th>
<th>No. of misclassification</th>
<th>Recognition rate (%)</th>
<th>time of recognition (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2500</td>
<td>2495</td>
<td>5</td>
<td>99.81%</td>
<td>0.93</td>
</tr>
<tr>
<td>5000</td>
<td>4976</td>
<td>24</td>
<td>99.52%</td>
<td>1.86</td>
</tr>
<tr>
<td>7500</td>
<td>7445</td>
<td>55</td>
<td>99.27%</td>
<td>2.85</td>
</tr>
<tr>
<td>10000</td>
<td>9909</td>
<td>91</td>
<td>99.09%</td>
<td>3.86</td>
</tr>
</tbody>
</table>

CONCLUSION

In this paper, the training samples are obtained by pre-processing of the original sample images primarily from the field of computer vision, especially the deep learning. The sample dataset reading method greatly shortens the duration of convolutional network training. By combining the computer vision with the classical deep learning tool (CNN), an excellent recognition rate can be attained effectively for characters with hard-to-distinguish features. The experiments show that the deep learning-based recognition can reach an accuracy of over 99%. Besides, the training time is also shortened greatly by changing the way data is read. The application of methods based on such data reading mode and deep learning is expected to be ever mature for the recognition of various more hard-to-distinguish characters.

ACKNOWLEDGMENT

This work was supported by the Science and Technology Project of Guangdong Province under Grand No. 2017B010132001 and 2016A010101038.

REFERENCES