An Encoder-Decoder Neural Network for Indefinite Length Digit Sequences in Natural Scene Recognition

Tong Zhang¹, Erhan Zhang², Hanfeng Zhang³, Feihong Shen⁴, Dongwei Guo⁵,* and Kaichao Lin⁶

¹School of Software, Jilin University, Changchun, Jilin 130012, China.
1277535055@qq.com
²School of Software, Jilin University, Changchun, Jilin 130012, China.
353075371@qq.com
³School of Software, Jilin University, Changchun, Jilin 130012, China.
zhanghf2048@gmail.com
⁴School of Software, Jilin University, Changchun, Jilin 130012, China.
feihong_jlu@QQ.com
⁵School of Software, Jilin University, Changchun, Jilin 130012, China.
130906800@qq.com
⁶School of Software, Jilin University, Changchun, Jilin 130012, China.
1785539271@qq.com
*Corresponding author’s e-mail: 130906800@qq.com

Abstract. Extracting text information from raw images has always been a hot and difficult problem in computer vision research. Due to blurred image, uneven illumination and complex backgrounds, etc., the recognition of natural scene digit is difficult to achieve desired results. In this paper, an encoder-decoder neural network model is proposed to solve the problem of recognition of indefinite length digit sequences in natural scene. The encoder is convolutional neural network (CNN), and the decoder is long short-term memory (LSTM). The encoder accepts a fixed-format image and outputs a fixed-length feature vector; the decoder accepts the feature vector and outputs a predictive sequence of indefinite length. The verification based on Google Street View house number Dataset (SVHN) shows that our method has a good performance. After 10 hours of training, whose accuracy is 96.57%.

1. Introduction
Recognition of printed and handwritten characters in documents can be regarded as a problem that computer vision has solved and has been widely used in industry and academia [1]. However, character recognition in complex natural scene is still a challenging puzzle. Pictures in natural scene are not as clear as the scanned documents with clean backgrounds. The character image in the natural scene is limited for the shooting conditions and is susceptible to the natural environment, which may cause problems such as image blurring, shifting, twirling and over, under or unevenly illuminated. Character recognition in natural scene can provide more accurate positioning services for a series of applications such as intelligent transportation and driverless car, which makes the related researches even more
important.

Due to the complex image background and various noises of characters in natural scene, the method of manually extracting features cannot achieve a good effect. The rise of deep learning represented by convolutional neural network (CNN) has brought breakthroughs to this problem [2].

In the study of the recognition of single digit in natural scene, Qiang Guo [3] used Hidden Markov Model to process the changes of the sequence and constructed a CNN-HMM hybrid model. The model is tested on the single digital version of google street view house number dataset (SVHN) with an accuracy rate of 81.07%. Ma [4] and others proposed a method that combined CNN and SVM. After 7 hours of training, the training accuracy reached 90.35%. The time was greatly shortened but the accuracy was still not high. Pierre Sermanet [5] et al. studied multi-stage characteristics in CNN and improved the traditional convNet architecture using LP pool, with a recognition rate of 94.85%. Ma [9] et al. used the WMF-CNN model, and the recognition rate of single digital version of SVHN dataset reached 95.6%. Wang Qiang [6] et al. combined principal component analysis (PCA) and CNN to achieve a recognition rate of 95.10%. It’s a great breakthrough, but it requires too much time cost, which is about 1 week.

In the study of the recognition of multiple digits in natural scene, Zhong Ju-ping [11] et al. used the improved LeNet-5 model. They define a sequence of numbers up to 5 digits in length and classify them in parallel using 5 SoftMax classifiers. The accuracy of the multi-digit version of the SVHN dataset is 89.14%.

Aiming at the problem of low recognition rate of multiple digit in natural scene, an encoder-decoded neural network is proposed. The encoder is a convolutional neural network (CNN) and the decoder is a long short-term memory (LSTM). The feature vectors extracted by CNN are input to LSTM. LSTM has good performance in solving sequential model problems, so the results are output by LSTM. After 10 hours of training, the recognition rate for the multi-digit version of the SVHN dataset is 96.57%.

### 2. The Design of Encoder-Decoder Neural Network

#### 2.1 A subsection

In previous studies on the recognition of indefinite digit sequences in natural scene, the features vector mapped by the last layer of CNN is directly sent to the classifier. It was not considered that the digits in the image have their natural order - arranged from left to right. We concatenate CNN with LSTM, which can predict digit sequence more efficiently.

In this paper, an encoder-decoder neural network model is proposed. The overall design of encoder-decoder neural network is shown in Figure 1. Below Image Embeddings is encoder and above is decoder. Our model can recognize indefinite length digit sequences in natural scene. The encoder accepts a fixed-format image and outputs a fixed-length feature vector; the decoder accepts the feature vector and outputs a predictive sequence of indefinite length.
2.2 Encoder

The encoder is a convolutional neural network we built. CNN is a network structure of local perception and weight sharing. It produces fewer trainable parameters than other networks, reducing learning complexity. Moreover, the CNN is highly invariant to image translations, tilts, scales and other forms of deformation.

The typical structure of a CNN consists of input layers, convolution layers, pooling layers, fully connected layers and output layers. The inputs are three-channel images; The function of the convolution layer is to extract features; The function of the pooling layer is to compress features and reduce computational complexity; The fully connected layer learns the global features and integrates the previously obtained local features; The last layer is the output of the network.

The CNN we constructed consists of input layer, 5 convolution hidden layers, 2 fully connected hidden layers and output layer. Each convolution hidden layer contains convolution, normalization, activation, maximum pooling and dropout operations. The filter size is $5 \times 5$, and its stride is 1, the pool size is $2 \times 2$, and its stride is 2. The padding strategy for both convolution and pooling operations is ‘same’.

After the 5 convolution hidden layers, the feature map is flattened to a vector. Input this vector into the fully connected hidden layer. Each fully connected hidden layer has 768 neurons. The output layer outputs a feature vector that is Image embeddings shown in Figure 1. The size of Image embeddings is $512$.

Figure 1. The overall design of encoder-decoder neural network.
2.3 Decoder

The function of decoder is to predict digit sequences from the feature vectors extracted by encoder. Decoder is long short-term memory (LSTM). LSTM is a kind of recurrence neural network (RNN). LSTM have long-term and short-term memory, usually used for sequence model tasks such as speech recognition and machine translation. LSTM was first proposed by Hochreiter S [11] and others in 1997, mainly to solve the gradient disappearance and gradient explosion problems in long sequence training. LSTM has better performance than ordinary RNN. Figure 2 shows a typical LSTM structure [8].

![Figure 2. Typical LSTM structure.](image)

The decoder we built using LSTM is shown in the top half of Figure 1. Input Image embedding, the output layer of the encoder, and then input the processed digit embedding one by one. $S[0]$ to $S[n-1]$ represent each digit embedding, and $O[1]$ to $O[n]$ represent the output of the LSTM network, which are a vector of a length of 64.

The output of the LSTM is computed by the fully connected layer to obtain the predicted components for each digit class. The prediction components corresponding to each category are normalized by the SoftMax function, whose formula is as follows:

$$y_i = \frac{e^{\hat{y}_i}}{\sum_{j=1}^{n} e^{\hat{y}_j}}$$

$y_i$ is the prediction probability for the next digit. Since the first digit embedding is the start mark, the predicted value of the first digit is obtained after $O[1]$ is processed. Similarly, the $O[n]$ can be used to get the end mark through the aforementioned process.

2.4 Loss function

The prediction probability of the next digit is recorded as $p$. In the case of batch training, when calculating the loss function, the actual digit of a batch is converted into a one-hot form with the same size as $p$. The loss value of a batch is the mean of the cross entropy of the $p$ and the actual digit.

The formula for calculating cross entropy is as follows:

$$loss = -\sum_{i=1}^{n} y_i \log(\hat{y}_i)$$

$\hat{y}_i$ is the prediction probability of the i-th category. The total loss value of one batch is:
We use gradient descent method to minimize the loss value and improve the prediction accuracy.

3. Experiment

3.1. Introduction to the dataset
The Street View House Number SVHN (SVHN) dataset is derived from the Google Street View house number and is a real-world image dataset used primarily to develop machine learning and image recognition algorithms. The SVHN dataset is divided into two versions, a single digit and a sequence of digits, each of which contains three subsets of the training set, the test set, and the extra set. Each subset contains house number pictures and manually annotated picture information (including the sequence of digits and the location of each number). The training set contains 33,402 images, the test set contains 13,068 images, and the extra set contains 202,353 images.

3.2. Data preprocessing
In a sample experiment, our training set is 90% randomly extracted from the training set and the extra set in the SVHN dataset, in addition, the verification set is the remaining 10%. Therefore, the size of the training set can be considerably enlarged. The test set for SVHN is used as our test set.

Preprocessing is mainly divided into preprocessing of pictures and preprocessing of labels.

Preprocessing the image: First, find a rectangular box that can contain all the digits in the image according to the annotation document. Find the center \( c \) of the rectangle. Set \( l \) and \( w \) is the length and width of the rectangle, the side length value of the square box \( s = \max(l, w) \). The center of the square box is \( c \). Thus we get a square box containing all the numbers in the picture. Expand this square box by 30%, and finally crop the image, leaving only the image in the border (including border), and resizing the image to \( 64 \times 64 \).

Use a square box instead of a rectangle to avoid image distortion.

In training, in order to increase the diversity of the samples and prevent over-fitting, we used a method of randomly trimming the image to \( 54 \times 54 \) and randomly adjusting the brightness, saturation, hue, and contrast.

Label preprocessing: We set 10 as the start sign and 11 as the end sign. In the experiment, we need input sequence: tag with start tag; target sequence: tag with end tag; at the same time, we use mask to record the length of target sequence. A typical label preprocessing is shown in Figure 3.

3.3. Training process
In the training, the batch size is adjusted to 32, and the dropout rates of CNN and LSTM are 0.2 and 0.3. The recognition accuracy is checked once every 1000 iterations, and the parameters are recorded for statistical analysis. The maximum accuracy is initially set to 0.0. If the current accuracy is greater than the maximum accuracy, the current accuracy is updated to the maximum accuracy. If the maximum accuracy rate is not updated more than 100 times, it is considered that the accuracy rate reaches or
approaches the maximum value at this time. At this time, the training is over, and redundant training is meaningless.

The exponential decay learning rate is used in the training so that the learning rate changes according to the following formula:

$$\text{decayed \_learning \_rate} = \text{learning \_rate} \times \left(\frac{\text{global \_step}}{\text{decay \_steps}}\right)^{\text{decay \_rate}}$$

The initial learning rate of the experiment is 0.1, and the learning rate is attenuated to 0.9 at each iteration. In order to make the model more stable in the later stages of training, a high learning rate is initially set to quickly reach a relatively good solution, and gradually reduced afterwards. The learning rate changes are shown in Figure 4.

![Figure 4. The learning rate changes](image)

3.4. Experiment results

The environment is: GPU: NVIDIA GeForce GTX 1050 Ti, memory 4G; CPU clock rate at 2.50GHz; memory 8G; 64-bit Microsoft Windows 10 operating system; using TensorFlow open source library, Python programming language.

After 7 hours training, the accuracy rate reached 96.5%. After that, the accuracy rate fluctuated between 96.3% and 96.6%. The accuracy reached a maximum value of 96.57% after training for 10 hours.

![Figure 5, 6. Loss value and accuracy change](image)

We compared our model with HOG[7], K-means[7], Hidden Markov Model (HMM) and CNN mixed model[3], LeNet-5 model combined with SVM[4], multi-stage feature learning combined with improved version of LP pool CNN model[5], improved CNN model for weighted multi-layer feature fusion[9], improved CNN model with principal component analysis (PCA)[6] and improved LeNet-5 model[10]. The difference on recognition accuracy among them are shown in Table 1.
Table 1. Classification accuracy of different methods

| Methods               | Accuracy (%) |
|-----------------------|--------------|
| HOG                   | 85.00        |
| K-MEANS               | 90.60        |
| CNN-HMM               | 81.07        |
| LeNet5-SVM            | 90.35        |
| Convnet/MS/L4         | 94.85        |
| WMF-CNN               | 95.60        |
| PCA-CNN               | 95.10        |
| Improved LeNet-5      | 89.14        |
| our method            | 96.57        |

The normalized confusion matrix of the experimental results on the verification set is shown in Figure 7. The abscissa indicates the prediction classification, and the ordinate indicates the true classification in the experiment. 11 is the end flag.

Figure 7 Normalized Confusion Matrix of the Experimental Results of the Method in the Validation Set

Figure 8 shows some digit sequence pictures that we randomly selected in the test set, and tested under the condition that only the image is cropped. The predicted result is that all 6 pictures are successfully recognized, and the correct result is ranked first in the confidence level. It can be seen that the method has better predicted and the accuracy is higher.
Figure 8. Randomly selected digit sequence pictures from the test set, correctly predicted.

This method predicts most of the SVHN datasets accurately, but there are still some errors. Figure 9 shows some of them. It can be seen that the image is too fuzzy, the background is complicated, and much interference still leads to prediction errors. In addition, vertical alignment of digits can also lead to errors in predictions. This may be due to the fact that such samples are too small in the training set to be adequately trained. Studying how to resolve these prediction errors is a possible improvement.

Figure 9. Some prediction errors.

4. Conclusion
In this paper, an encoder-decoder neural network model is proposed to solve the problem of recognition on indefinite digit sequences in natural scene. After 10 hours of training, the accuracy is 96.57%. Our model has shorter training time and higher accuracy than other models. In addition, because the network parameters of the model are few, the recognition speed is fast. Our models meet the real-time and accuracy requirements of future applications such as auto-navigation addressing and driverless car. The encoder-decoder neural network has the characteristics of accepting fixed-length sequences and outputting variable-length sequences. It also has a good effect on other character sequence recognition
problems.

Acknowledgments
This project is supported by Jilin University students' innovation and entrepreneurship project.

References
[1] Plamondon R. On-line and off-line handwriting recognition: A comprehensive survey[J]. (2000) IEEE Trans. Pattern Anal. Mach. Intell.
[2] JIAO Li-Cheng, YANG Shu-Yuan, LIU Fang, et al. Seventy Years beyond Neural Networks: Retrospect and Prospect [J]. (2016) Chinese Journal of Computers.
[3] Guo Q, Tu D, Let J, et al. Hybrid CNN-HMM Model for Street View House Number Recognition[C]. (2014) Computer View—ACCV Work.
[4] Ma Miao, Chen Fang, Guo Min, et al. A recognition method based on improved LeNet-5 for street view house numbers [D2016]. Journal of Yunnan University.
[5] Sermanet P, Chintal S, Lecunn Y. Convolutional Neural Networks Applied to House Numbers Digit Classification[C]. International Conference on Pattern Recognition (2012). IEEE:3288-3291.
[6] Wang Qiang. Research on Character Recognition Based on CNN [D] (2014). Tianjin: Tianjin Normal University.
[7] Nenzer Y, Wang T, Coates A, et al. Reading Digits in Natural Images with Unsupervised Feature Learning[C], (2011). NIPS Workshop on Deep Learning and Unsupervised Feature Learning:1-9.
[8] Graves A Long Short-Term Memory[M]// Supervised Sequence Labelling with Recurrent Neural Networks. Springer Berlin Heidelberg, (2012).
[9] Ma Miao, Liu Lin, Chen Yu-Li. A WMF-CNN model for street view house number recognition [J] (2018). Journal of Yunnan University (Natural Sciences Edition).
[10] Zhong Ju-ping, Gao Jing, Li Jun. Multi-Digit Number Recognition from SVHN in Natural Scene Based on Convolutional Neural Network [J] (2018). Modern Computer: Mid-term Journal.
[11] Hochreiter S, Schmidhuber, Jürgen. Long Short-Term Memory[J]. (1997) Neural Computation.