Machine Vision and Applications (2018) 29:861–871
https://doi.org/10.1007/s00138-018-0942-y

ORIGINAL PAPER

Scene text recognition using residual convolutional recurrent neural network

Zhengchao Lei[1] · Sanyuan Zhao[1] · Hongmei Song[1] · Jianbing Shen[1]

Received: 24 October 2017 / Revised: 26 March 2018 / Accepted: 12 May 2018 / Published online: 16 June 2018

Abstract
Text is a significant tool for human communication, and text recognition in scene images becomes more and more important. In this paper, we propose a residual convolutional recurrent neural network for solving the task of scene text recognition. The general convolutional recurrent neural network (CRNN) is realized by combining convolutional neural network (CNN) with recurrent neural network (RNN). The CNN part extracts features and the RNN part encodes and decodes feature sequences. In order to improve the accuracy rate of scene text recognition based on CRNN, we explore different deeper CNN architectures to get feature descriptors and analyze the corresponding text recognition results. Specifically, VGG and ResNet are introduced to train these different deep models and obtain the encoding information of images. The experimental results on public datasets demonstrate the effectiveness of our method.

Keywords Residual convolutional recurrent neural network · Scene text recognition · Convolutional neural network · Recurrent neural network · Residual network

1 Introduction

Scene text recognition refers to recognize the text from natural scene images and output the corresponding text characters. As an important direction of text detection and recognition, scene text recognition is widely applied in the fields of intelligent license plate recognition [1], road marking recognition [2] and so on. Nevertheless, there are some factors lead to the difficulty of scene text recognition. First of all, scene texts are varied in terms of font, color, and size. Secondly, non-text objects in the scene, such as landmarks and railings, may overlap with the scene text and act as a part of the text. Moreover, scene text images are affected by light, angle, etc, which makes the scene text images contain noise, low resolution, low illumination or partial occlusion. Therefore, the scene text recognition is a certain challenge and worthy of study.

With the development of deep learning, varieties of methods have been proposed to address the above challenges. In general, methods of scene text recognition can be roughly divided into two categories. One mainly concentrates on developing a powerful character classifier and then connects the corresponding characters to text. However, these methods merely concerned individual character and ignored the context of scene text, leading to lower recognition accuracy. The other one focuses on the contextual information, which recognizes scene text as a whole in various ways, and gradually becomes the new trend in recent years. Rodriguez et al. [3] utilized label embedding to match between strings and images, by passing pre or post-processing operations. Jaderberg et al. [4] proposed a CNN model with 8 layers to recognize the entire scene text. Shi et al. [5,6] exploited the combination of CNN and RNN to take the contextual information into account. However, these methods only use shallow neural networks to extract the features ignoring the deeper information. The layer number of the neural network may affect the recognition accuracy of scene text as well.

In our work, we combined CNN with RNN to recognize scene text, which is similar to CRNN [5,6]. The CNN part

✉ Sanyuan Zhao
zhaosanyuan@bit.edu.cn

Zhengchao Lei
leizhengchao@bit.edu.cn

Hongmei Song
songhongmei@bit.edu.cn

Jianbing Shen
shenjianbing@bit.edu.cn

[1] Beijing Key Lab of Intelligent Information Technology, School of Computer Science, Beijing Institute of Technology, Beijing 100081, People's Republic of China

is used to extract the features of images and encode them to feature sequences, and the BLSTM part is used to decode the corresponding sequences to labeled sequences. In the experiments, we exploited different depths of CNN for better encoding information of images, including VGG16 and VGG19 [7]. By comparing the accuracy of these two models, the degradation problem occurs that VGG19 doesn't perform better than VGG16 as expected, while this was explained in [8], leading to performance decline because of the vanishing gradient. We incorporated the residual network [8] to deal with the degradation problem. ResNet has been widely used in saliency detection [9–12], object detection [13–15], object co-segmentation [16–18], and scene classification [19].

The main contributions of our work are threefold:

(1) We explored various CNN architectures to represent the images and analyze their effectiveness including VGG16, VGG19, ResNet34, and ResNet50.
(2) We also utilized different depths of RNN structures, such as BLSTM and LSTM, to learn the sequential information extracted by CNN part. We rearranged the image in training set according to their aspect ratio to speed up the convergence.
(3) The experimental results demonstrated that by adopting deep semantic level information from Residual Network and BLSTM, our proposed residual convolutional recurrent neural network performed better than the state-of-the-art methods.

2 Related work

Numerous studies on scene text recognition have been carried out in recent years. Most of these methods mainly focus on classifying the characters of scene text and connect them to the final recognized text. Wang et al. [20] proposed a character classifier using HOG features with random ferns. In [21], Wang et al. detected and classified the characters with a two-layer CNN. Bissacco et al. [22] proposed a PhotoOCR system for the character recognition which employed a five-layer deep neural network. Neumann and Matas [23] detected and classified characters with an oriented stroke which was limited by the low-level features. Lee et al. [24] introduced a mid-level representation of characters by proposing a discriminative feature pooling. Similarly, Yao et al. [25] proposed the mid-level Strokelets to describe the parts of characters. All of these methods only consider the individual character of scene texts and may affect the recognition results.

The other methods take the contextual information of characters into account. Almazan et al. [26] proposed to predict label embedding vectors from input images. This method jointly embedded both word images and text strings into the common features. Jaderberg et al. [4] addressed text recognition with a 90k-class convolutional neural network, in which each class corresponded to a word. While this model was strictly constrained by the predefined dictionary, which makes it unable to recognize novel text. In [27], Jaderberg et al. proposed a method for unconstrained text recognition by utilizing a CNN with structured output layer. However, it was difficult to identify multiple words accurately for lacking of distinguishing non-character space. Su and Lu [28] used a sequence of HOG [29] descriptors to represent the images and predicted the corresponding character sequence with RNN. The method of CRNN [5] was an end-to-end neural network architecture which combined CNN with RNN. Shi et al. [30] proposed Robust text recognizer with automatic rectification (RARE), a recognition model that was robust to irregular texts. RARE was a specially designed deep neural network, which consisted of spatial transformer network (STN) and sequence recognition network (SRN).

Although a lot of methods have been introduced for scene text recognition, different size of neural network is rarely discussed, while it plays a crucial role in the deep learning models. Moreover, features in deep layers [31,32] will carry more information of scene text images than features only from shallow layers.

3 CRNN for scene text recognition

CRNN was firstly proposed by Shi et al. [5]. There are three components in this method, including the convolutional part, the recurrent part and a transcription layer. At the bottom of CRNN, the convolutional part consisted of convolutional and max-pooling layers from a standard CNN model with removing fully connected layers [7]. This component was used to extract a sequential feature representation from an input image. Specifically, the authors simply used 7 convolutional layers and 4 max-pooling layers and then tweaked the size of pooling window. Additionally, they inserted batch normalization layers [33] after the 5th and 6th convolutional layers, respectively.

Before [5], many researchers have explored CNN models for different kinds of visual recognition tasks such as [34,35]. The previous method [4] also adopted CNN to scene text recognition. These methods extracted global representations of the whole image by CNN and collected local features for recognizing each component of sequential objects. However, CNN was not appropriate for sequential object since the length of sequential objects is varied, and the input images should be scaled to fixed size.

Similar to CRNN, deep-text recurrent network (DTRN) [36] also introduced recurrent layers to make up the deficiency of CNN. Compared to CNN, RNN is capable of extracting contexts of sequential objects and back-

propagating error differentials to input. Moreover, RNN plays more advantages in processing sequences with arbitrary lengths. However, the traditional RNN suffers from the vanishing gradient problem [37] which limits the range of context to be stored and burdens the training performance.

Long–short-term memory (LSTM) [38,39] is a typical RNN part to address the above problem. Each LSTM block composes a memory cell, an input gate, an output gate and a forget gate. The memory cell is used to store past contexts, and the input and output gate allow the cell store contexts for a long period of time. The forget gate will clear the memory cell on a certain probability. Therefore, LSTM is able to capture long-range dependencies of image-based sequences. Nevertheless, contexts from both directions are expedient for complementing in sequential objects, while LSTM simply uses past contexts due to its directionality. Therefore, the authors follow [40] to combine two LSTMs into bidirectional LSTM (BLSTM). Furthermore, they construct multiple BLSTMs with 256 hidden neurons in each layer.

The component of transcription layer is adopted to create a mapping model between output sequence of RNN and label sequence. The label sequence with the highest probability is the final result of recognized text. The lexicon-free and lexicon-based transcriptions are of two transcription layers, where the lexicon is a set of label sequences of which the prediction is restricted to, e.g., a spell checking dictionary. Specifically, predictions are made without any lexicon in the lexicon-free mode while they are made with the lexicon in the lexicon-based mode. In CRNN model, connectionist temporal classification (CTC) layer proposed by Graves et al. [41] is adopted to define the conditional probability.

4 The improvements on the basis of CRNN

The architecture of our work is shown in Fig. 1. We utilize CNN [7,8] to extract image features and encoded them into feature sequences. Then, we decode above sequences through BLSTM [40] and output label sequences. In the final, label sequences are mapped to text by CTC [41] layer and the recognized text is obtained. We build two different models, which exploits VGG [7] and residual network [8] as our CNN part, respectively. Each model follows with one BLSTM layer as the RNN part.

4.1 CNN for encoding

In the CNN part, we introduce deeper neural networks to enhance the accuracy rate. Specifically, we explore different deeper feature descriptors of VGG16, VGG19, ResNet34, and ResNet50, respectively, and make comparisons among the results.

Since the text direction of images in the dataset is horizontal, we can downsample several times in vertical direction until the height of feature map is reduced to 1, and slightly in horizontal direction. This is because the excessive downsampling may lead to overlap of encoding sequences between two adjacent characters, thereby reducing the character encoding quality. The output of final CNN part is the feature map with height 1 which is easily converted to a feature sequence.

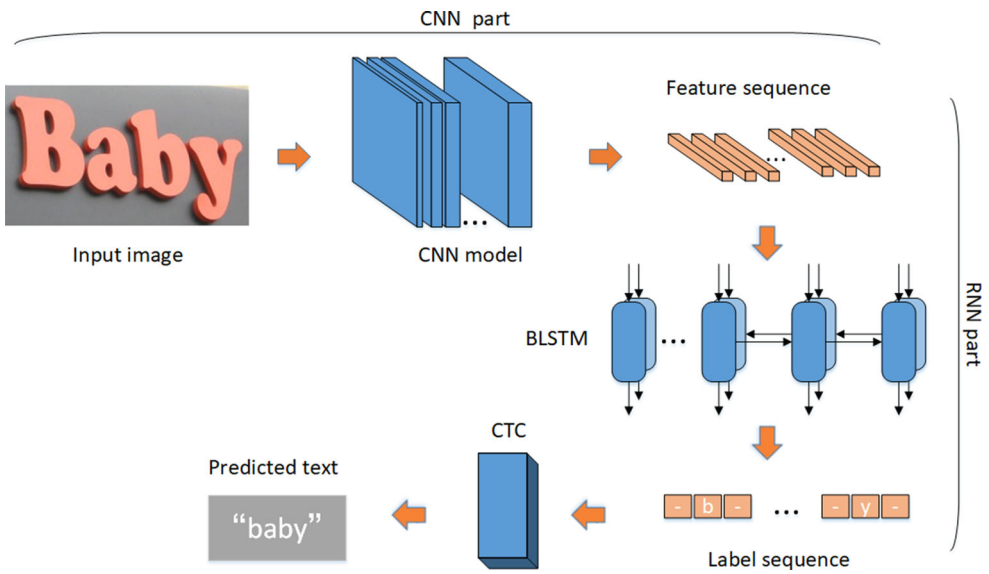

Fig. 1 The architecture of our work. We first utilize CNN to extract features and encode them into feature sequences. Then, we decode the feature sequences through BLSTM and output label sequences. The labeled sequences will be mapped to words by CTC layer. Finally, we obtain the recognition results

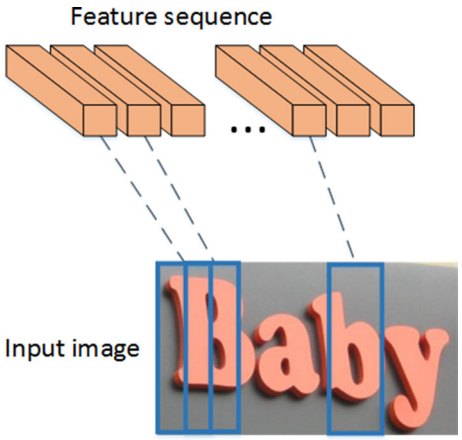

Fig. 2 The corresponding relationship between feature sequence and input image

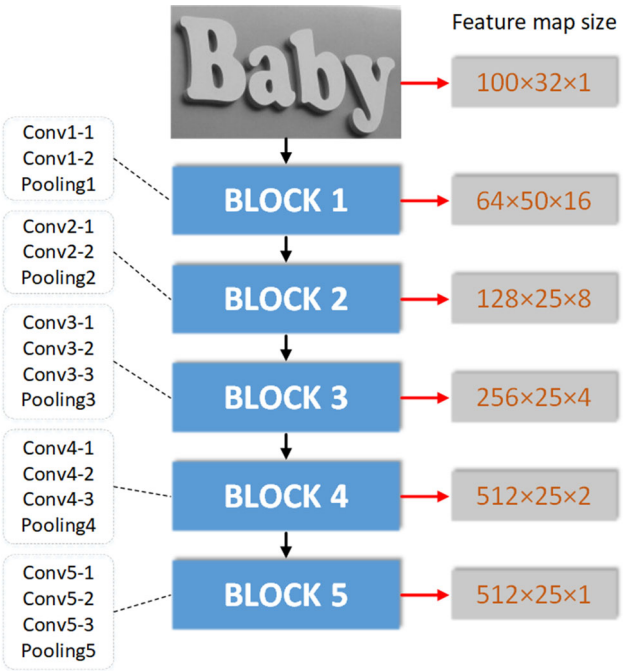

Fig. 3 The architecture of VGG network and the size of its corresponding feature map in our CNN part

The corresponding relationship of feature sequence and input image is shown in Fig. 2.

Firstly, we conduct VGG16 model for discussion. Figure 3 shows the architecture of VGG16 and corresponding feature map size in our CNN part. Moreover, we add one batch normalization [33] layer after each convolutional layer. The VGG16 model does not change the size of feature map in the convolutional layers. However, the width and height of feature map are reduced to half in the max-pooling layers. There are totally 5 max-pooling layers in VGG16, which leads the width and height of feature map reduced to 1/32 of the original size. In our experiments, we keep the first two

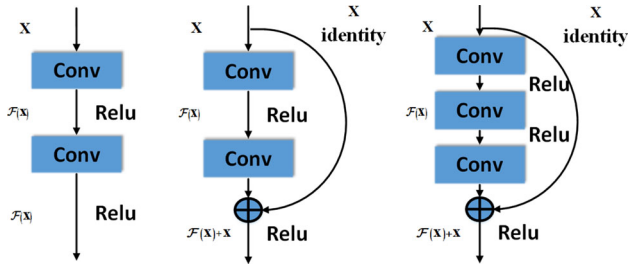

Fig. 4 The difference of structure in convolution layers between VGG and ResNet. The left is the structure in VGG, the center is the block used in ResNet18 and ResNet34, while the right is the block used in ResNet50, ResNet101 and ResNet152

max-pooling layers unchanged and still use 2×2 kernel with 2×2 stride for pooling. Then, we set 1×2 stride in the following three max-pooling layers to ensure the feature map of horizontal direction would not be downsampled. Thereby the height of final feature map is reduced to 1 pixel, and the width is reduced to a quarter of original width. The using of first two max-pooling layers is to narrow down the feature map since it reduces the size of feature map earlier and decreases calculations of the model. Finally, VGG16 outputs a $512 \times 25 \times 1$ feature map. According to the corresponding relationship of feature sequence and input image shown in Fig. 2, each vector with the length of 512 corresponds to a few columns of the image. Thus, we can convert the 512×25 matrix to a sequence with the length of 25, as the result of encoding information of input image.

After conducting VGG16 as the CNN part, we try to use a deeper network VGG19, hoping to improve the recognition ability of our model. The difference from VGG16 is to add a convolutional layer in each last three blocks. Compared to the model using VGG16, VGG19 can not improve the recognition accuracy as expected, even a slight decline (see Table 2). This is the so-called degradation problem while simply increasing the number of convolutional layers is not able to improve the performance of the features extracting and encoding. Therefore, we decide to introduce the residual neural network [8] in our model. The difference of structure in convolution layers between VGG and ResNet is shown in Fig. 4.

The output vector y of VGG can be defined as:

$$y = \mathcal{F}(x, W) \tag{1}$$

where x denotes the input vectors of layers and W refers to the parameters to be learned. The function $\mathcal{F}$ is the operation that mapping x–y. Furthermore, x should be added after the operation $\mathcal{F}$ as mentioned in ResNet [8]:

$$y = \mathcal{F}(x, \{W_i\}) + x \tag{2}$$

where W_i means the parameters of ith convolutional layers to be learned.

As shown in Fig. 4 (center and right), a typical structure of ResNet contains two or three convolutional layers. Meanwhile, $\mathcal{F}(x, \{W_i\})$ could represent multiple convolutional layers in practical use, and it is the residual mapping to be learned.

Compared to ResNet34 and ResNet50, ResNet18 would not represent the image feature perfectly for shallow architecture. ResNet101 and ResNet152 require a significant amount of computing resources. Therefore, we introduce ResNet34 and ResNet50 in our experiments. Similar to VGG, ResNet downsamples the width and height of input image at the procedure of capturing features. In order to obtain the same width and height of feature map, we make some changes in ResNet. ResNet utilizes 7×7 kernel in the first convolution layer to halve the height and width of input image. The max-pooling layer halves the height and width as well. In the following blocks, the first convolutional layer of each block, except for the first block, with 2×2 stride is also downsampling the feature map at a half, We change 2×2 stride of the first layer convolution layer to 1×2 stride in each block. In addition, the procedure of width downsampling is removed, and height downsampling is kept in each block. Therefore, we still downsample the width and height of input image two times and five times for ResNet, respectively.

As a result, the final output of ResNet contains the same feature sequence with 25×1 to VGG. The difference between them is the encoded sequence length. ResNet generates the encoded sequence with the length of 512 for ResNet34 and 2014 for ResNet50, while VGG generates the sequence with 512. Figure 5 shows the architecture of ResNet50 and the corresponding feature map size, which replaces the previous CNN part. For ResNet34, the architecture is similar to Fig. 5, while the block should be replaced by the block shown in Fig. 4 (center). The sequence length of each block in ResNet34 is 64, 128, 256 and 512.

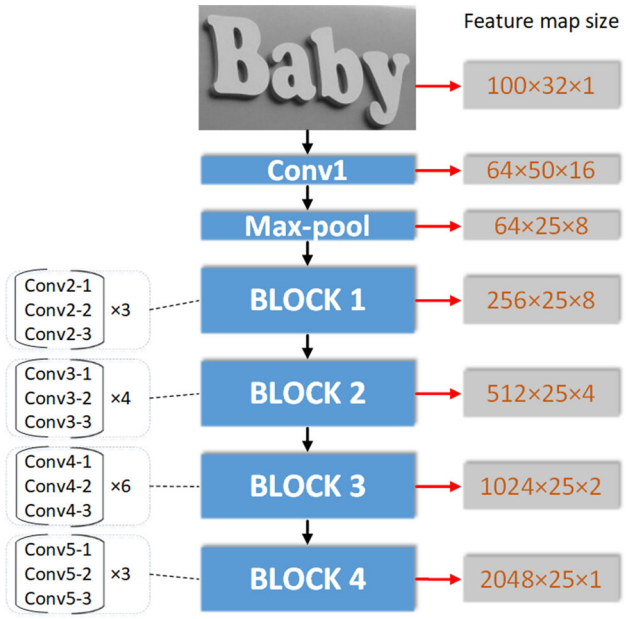

Fig. 5 The architecture of ResNet50 and the size of its corresponding feature map including the input image in our ResNet part

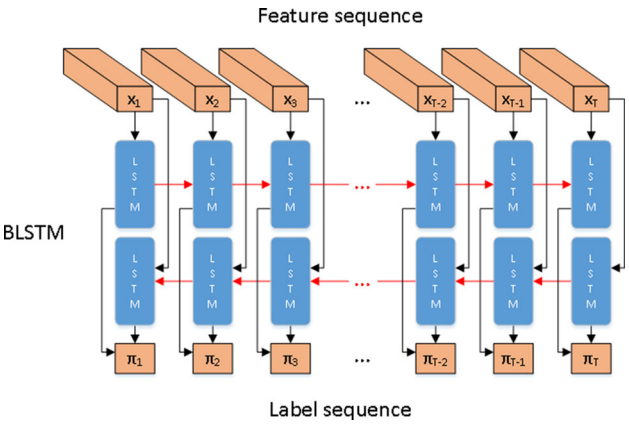

Fig. 6 The structure of our BLSTM

4.2 BLSTM for decoding

After extracting the feature of input image by VGG or ResNet model and encoding the feature into corresponding sequence, we utilize the LSTM layer to decode the feature sequence. In our experiments, we introduce both LSTM and BLSTM as our RNN part. Since BLSTM takes both the sequence before current time and the sequence after current time into account. Therefore, each character of the result in scene text recognition will consider the context before and after, which can effectively reduce the error rate in text recognition. The experimental results also show effectiveness of BLSTM compared to LSTM in Table 3

Different from CRNN [5], we use only one BLSTM layer to decode the feature sequence because it gets a better result in scene text recognition and reduces computational cost similar to DTRN [36]. In detail, we set the number of hidden neurons to 512 in our BLSTM. Figure 6 shows the structure of BLSTM used in our experiments. The number of hidden neurons determines the length of cell state. The longer of states, the more context contains, then the higher accuracy of text recognition would be. However, increasing the number of hidden neurons will increase the computational cost, while the context information is limited. Therefore, it requires us to explore a suitable number of the hidden layer neurons in our experiments.

In order to select an appropriate number of hidden neurons, we train our models with hidden neurons 128, 256, 512 and 1024, respectively, and draw a comparison in a number of test sets. Finally, we set the number of hidden neurons to 512. The result is shown in Table 1.

Table 1 Accuracy comparison with the different numbers of hidden neurons in BLSTM

Model	Structure	IC03 (%)	IC13 (%)	SVT (%)	IIIT5K (%)
Model A	VGG16 + BLSTM (128)	88.2	83.3	79.2	80.4
Model B	VGG16 + BLSTM (256)	89.0	85.4	80.5	78.4
Model C	VGG16 + BLSTM (512)	89.6	84.7	80.7	81.2
Model D	VGG16 + BLSTM (1024)	89.7	85.3	80.6	80.5

4.3 Label sequence mapping

Essentially, CTC [41] layer is a mapping relationship between output sequence and final text, that is mapping the output sequence of BLSTM to the recognized text. For example, the input sequence of BLSTM is $x_1x_2x_3x_4x_5x_6$ and the labels are 'a' or 'b', it is possible to output "a-a-b-" or "-aa-b-" by introducing blank label '-' through BLSTM. However, they are the same as "ab" in CTC layer, which means "a-a-b-" and "-aa-b-" are all can be mapped to "ab". Meanwhile, "-aa-b-" can also be mapped to "aab". As we can see that the same input sequence can be mapped to multiple output label sequences in the CTC layer. As a result, we choose the final output label sequence according to maximum likelihood. The probability of output label sequence can be calculated by

$$\begin{aligned} p(l \mid x) &= \sum_{\pi \in F^{-1}(l)} p(\pi \mid x) \\ p(\pi \mid x) &= \prod_{t=1}^{T} y_{\pi_t}^t \end{aligned} \tag{3}$$

where x denotes the input sequence with the length of T. π means the output sequence with blank labels. l is the final output label sequence. The mapping function F defines the relationship between sequence π to label l. $y_{\pi_t}^t$ stands for the probability of having label π_t at time stamp t.

4.4 Training

Referring to the training method of [6] in our work, we resize each image in our training sets to 100×32 and train our model until it converges. Unlike [6], we resize the images only when the image width is larger than 100. At the same time, we remove the images of which aspect ratio is too large or too small since they are distorted and useless in our training step. Our objective is to minimize the negative log-likelihood of the conditional probability of ground truth. It is defined as the loss function L for the neural network:

$$\begin{aligned} L(S) &= \sum_{(x,z) \in S} L(x, z) \\ L(x, z) &= -\ln p(z \mid x) \end{aligned} \tag{4}$$

where $S = \{x, z\}$ is our training set, while x and z represent input sequence and ground truth sequence, respectively.

In order to optimize the learning rate in our training, we utilize ADADELTA [42] strategy with ϵ 1e − 6 and ρ 0.9. In RNN part, we apply back-propagation through time (BPTT) [43] to calculate the error differentials.

5 Experiments

5.1 Datasets

The training dataset is extremely important to train an excellent model in deep learning. It requires the dataset covers samples as comprehensive as possible, and various samples should be well distributed. Meanwhile, the size of dataset also affects the generalization capacity of models.

In the field of scene text recognition, synthesizing and simulating scene text images by computer is an effective way, since it is difficult to obtain large-scale datasets by collecting labeled text images in real scenes. In this paper, we utilize MJSynth as our training and test dataset. MJSynth is synthesized by Jaderberg [4,44] through a specific algorithm. This dataset combines the dictionaries of IC03-Full [45] and SVT-Full [20] by adding various fonts, background colors, textures, and noises. And it consists of about 9 million scene text images, where about 8 million images are used to train and one million used to validate. Each scene text image is labeled with a total of 62 characters, including 52 case letters and 10 Arabia numerals. Therefore, there consists of 36 labels, including 26 letters and 10 Arabia numerals.

Generally, there are two metrics to evaluate the performance in scene text recognition: One is to evaluate the performance in character level, and the other is in string level. Intuitively, the second metric is more rigorous. since the result is correct only each corresponding character is identified correctly. However, the metric in character level generally calculates the distance between the recognized text and ground truth, while the method with the least distance is considered to be the best. Therefore, we refer the second metric as our evaluation. The test datasets used in our experiments are as follows:

SVT [20] includes 249 images. We clip 647 images from the dataset according to the labels and treat them as our test set.

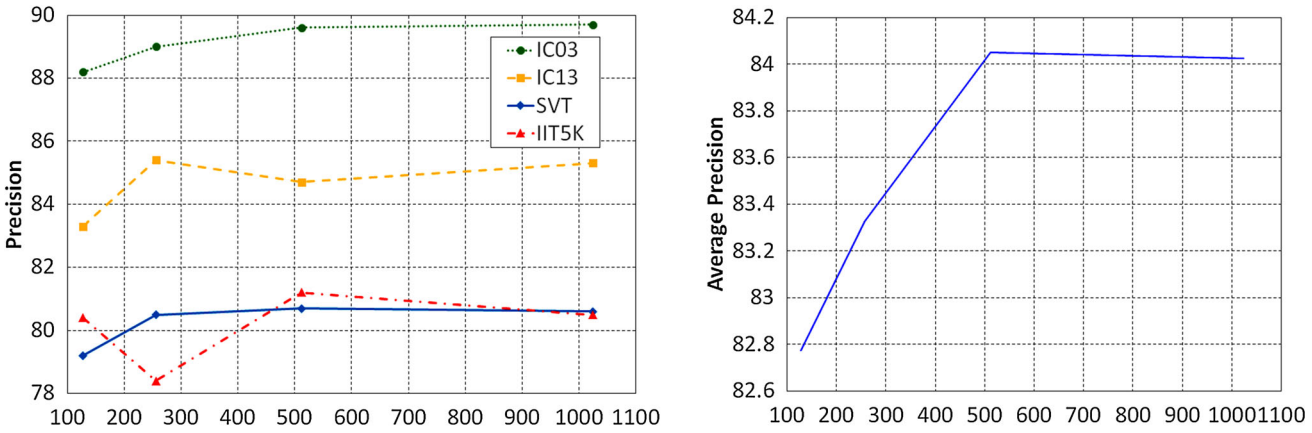

Fig. 7 The relationship between the number of hidden neurons and accuracy. The left is the relationship between the number of hidden neurons and accuracy in different datasets, while the right is the relationship between the number of hidden neurons and average accuracy in all test datasets including IC03, IC13, SVT and IIIT5K

IIT5K [46] contains 3000 scene text images. We use all of them for test.

IC13 [47] consists of 1095 scene test images. However, some of these images contain non-numeric and non-characters. We remove those images and eventually obtain 1015 images for test.

IC03 [45] contains 251 original text images. Firstly, we clip text patches from the original images according to the annotations. Then, the patches will be processed to remove the parts of non-numeric, non-characters parts, as well as the parts less than 3. Finally, 860 scene text images are obtained as the final test set.

5.2 Comparison of different numbers of hidden neurons in BLSTM

In order to keep the effectiveness of test results from distortion due to the resizing operation, we resize the images according to the original aspect ratio during our test step. The height of images is still kept in 32, while the width will be resized with the aspect ratio of 100×32 when the width is less than 100.

As mentioned in Sect. 4.2, we train our models with different numbers of hidden neurons in BLSTM. Table 1 shows the difference between hidden neurons 128, 256, 512, and 1024 on accuracy of scene text recognition combining with VGG16. The relationship between number of hidden neurons and average accuracy in IC03, IC13, SVT, and IIIT5k is shown in Fig. 7. From Fig. 7, we can see that the number of hidden neurons affects the accuracy in different datasets, and the average accuracy increases along with the increase in hidden neurons in the beginning. From Table 1, model B performs best in IC13 and model D performs best in IC03. However, model B is worse than model C with 0.6, 0.2 and 2.8% in IC03, SVT, IIIT5K, respectively, while model D is worse than model C with 0.1% in SVT and 0.7% in IIIT5K. Although model with 512 hidden neurons doesn't perform best in IC03 and IC13, the average accuracy tends to saturation and increasing number of hidden neurons would not enhance average precision significantly (see Fig. 7). On the contrary, it may lead to huge amount of computation. So we set 512 hidden neurons in our BLSTM.

5.3 Comparison of different improved models

After choosing the hidden neurons 512, we get four models as listed in Table 2 with their scene text recognition accuracy comparison. We combine VGG16, VGG19, ResNet34 and ResNet50 with one BLSTM layer in Model E, F, G, and H, respectively. We can see that the accuracy declines when we only simply increase the convolution layers from model E and model F except in SVT, while model F merely enhances the accuracy with 0.1%. Moreover, when we introduce ResNet as CNN part. As shown in Table 2, Model G with ResNet34 performs roughly equal to Model E, while the accuracy of model H improves significantly. We believe that simply increasing convolutional layers would not be effective in our tasks. In addition, it proves that the deeper neural network can get higher-dimensional feature descriptors than shallow network. The deeper feature descriptor can obtain better feature sequences of scene text images.

For the RNN part, we conduct experiments on LSTM and BLSTM listed in Table 3. Model E′ and F′ contain a LSTM layer after the CNN part. Compared to BLSTM, LSTM merely utilizing the unidirectional context information. As shown in Table 3, models with LSTM perform worse than models with BLSTM significantly. Meanwhile, we observe that models with LSTM consumes much more time to make

Table 2 Accuracy comparison with different CNN models

Model	Structure	IC03 (%)	IC13 (%)	SVT (%)	IIIT5K (%)
Model E	VGG16 + BLSTM (512)	89.6	84.7	80.7	81.2
Model F	VGG19 + BLSTM (512)	89.1	84.6	80.8	79.1
Model G	ResNet34 + BLSTM (512)	89.5	84.4	80.4	80.8
Model H	ResNet50 + BLSTM (512)	90.8	88.6	82.3	82.2

Table 3 Accuracy comparison with different RNN models

Model	Structure	IC03 (%)	IC13 (%)	SVT (%)	IIIT5K (%)
Model E	VGG16 + BLSTM (512)	89.6	84.7	80.7	81.2
Model E′	VGG16 + LSTM (512)	85.1	81.3	79.3	76.7
Model H	ResNet50 + BLSTM (512)	90.8	88.6	82.3	82.2
Model H′	ResNet50 + LSTM (512)	86.0	83.1	80.6	75.5

Table 4 Accuracy comparison with improved models

Model	Structure	IC03 (%)	IC13 (%)	SVT (%)	IIIT5K (%)
Model I	VGG16 + BLSTM (512)	90.8	86.5	82.3	82.2
Model J	ResNet + BLSTM (512)	91.5	88.7	82.8	82.0

Table 5 Accuracy comparison with other state-of-the-art methods (Top three methods are highlighted in red, green, blue)

Algorithm	IC03 (%)	IC13 (%)	SVT (%)	IIIT5K (%)
Bissacco et al. [22]	–	87.6	78.0	–
Bhunia et al. [48]	–	82.31	–	–
Jaderberg et al. [4]	93.1	90.8	80.7	–
Jaderberg et al. [27]	89.6	81.8	71.7	–
Lee et al. [49]	88.7	90.0	80.7	78.4
CRNN (2015) [5]	89.4	86.7	80.8	78.2
CRNN (2016) [6]	91.9	89.6	82.7	81.2
Shi et al. [30]	90.1	88.6	81.9	81.9
Our model I	90.8	86.5	82.3	82.2
Our model J	91.5	88.7	82.8	82.0

it converge to the final result than BLSTM. Therefore, we adopt BLSTM as our final RNN part.

We choose VGG16, ResNet50 as CNN part and BLSTM as RNN part to do the following optimization. Firstly, we rank the images in training set according to the aspect ratio and choose a set of continuous images as a batch randomly. For all images in this bath, the height will be resized to 32 according to the middle aspect ratio of this batch. Then, the new training dataset is obtained and fed into Model E and H to train another two models. At last, we get two final models I and J listed in Table 4 with their scene text recognition accuracy. We can see that the accuracy increase to a different extent both of two models after optimization. Model J still performs better than model I 0.7, 2.2 and 0.5% in IC03, IC13 and SVT, respectively, while worse than Model I 0.2% in IIIT5K. This can be explained that Model J does not converge significantly in the training step. Model J still obtain better scene text results than Model H in other datasets, which proves that deeper neural network performs better than shallow neural network in scene text recognition.

5.4 Compare with state-of-the-art methods

The comparison with state-of-the-art methods is shown in Table 5. '-' means the paper did not provide the evaluation results on the datasets, like method [4,22,27,48] on IIIT5K. The accuracy highlighted in red means the best result among all the methods, while accuracy highlighted in green and blue means the second and third results, respectively. Jaderberg et al. [4] perform the best in IC03 and IC13, which transforms the recognition to multi-class classification and exploits CNN classifier to achieve it. Although our both two models do not work better than Jaderberg et al. [4] in IC03 and IC13, they get best the results in IIIT5k and SVT. More-

Fig. 8 Examples of misrecognized text. GT means the true label of scene text image. The top row shows the scene text recognized correctly, while middle row shows some scene text recognized by one model while misrecognized by another model. The bottom row shows some text images that cannot be recognized by both of our two models

over, CRNN(2016) [6] only modifies the training method of CRNN(2015) [5] and keep the same network architecture. As a result, it enhances the accuracy rate shown in the table. Model I performs best in IIIT5K. Model J also get better results than the basis of CRNN(2016) [6] in SVT and IIIT5K, which means our approach is still effective compared to basic CRNN. Therefore, introducing deeper neural network is an alternative method to enhance the accuracy rate, and it can obtain the deeper feature descriptors for scene text recognition tasks.

Figure 8 shows some examples of misrecognition by model I and model J. The misrecognized text is highlighted in red. In the case of some special scene text images, such as images with art font texts, low resolution, and poor illumination, our models still perform the best in the top row of Fig. 8. At the same time, the scene text can also be recognized by one model while misrecognized by another model with one or two false characters as shown in the middle row of Fig. 8. This might be the different descriptors of two models lead to this result. However, there are still drawbacks in our models for the recognition of similar texts such as 'o' and '0', 'I' and '1'. We list some misrecognized examples by our models in the bottom row of Fig. 8. The last scene image is recognized as "i0" and "io" by the model I and model J, respectively, while the ground truth is "10".

6 Conclusion

In this paper, we explored different deeper neural network models to get the deeper feature descriptor on the basis of CRNN, which combines CNN and RNN for scene text recognition. We first introduced VGG16 and VGG19 as the CNN part, respectively. In order to address the degradation problem, we conduct residual network as the CNN part. Moreover, we utilize one BLSTM layer in the RNN part and compare the scene text recognition results with the different number of hidden neurons. At last, we choose an appropriate model which integrates the advantages of both CNN and RNN to solve scene text recognition problem. Furthermore, the experimental results demonstrate that our method can obtain better results on four benchmarks than other state-of-the-art methods. The deeper CNN with deep features descriptor can improve the accuracy of scene text recognition effectively.

Acknowledgements This work was supported in part by the Beijing Natural Science Foundation under Grant 4182056. Specialized Fund for Joint Building Program of Beijing Municipal Education Commission.

References

1. Thome, N., Vacavant, A., Robinault, L., Miguet, S.: A cognitive and video-based approach for multinational license plate recognition. Mach. Vis. Appl. **22**(2), 389–407 (2011)
2. Kheyrollahi, A., Breckon, T.: Automatic real-time road marking recognition using a feature driven approach. Mach. Vis. Appl. **23**(1), 123–133 (2012)
3. Rodriguez, J., Perronnin, F.: Label embedding for text recognition. In: British Machine Vision Conference, pp. 5.1–5.12 (2013)
4. Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A.: Reading text in the wild with convolutional neural networks. Int. J. Comput. Vis. **116**(1), 1–20 (2016)
5. Shi, B., Bai, X., Yao, C.: An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition. arxiv (2015)

6. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Trans. Pattern Anal. Mach. Intell. **39**(11), 2298–2304 (2017)
7. Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv (2014)
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: International IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
9. Yao, X., Han, J., Zhang, D., Nie, F.: Revisiting co-saliency detection: a novel approach based on two-stage multi-view spectral rotation co-clustering. IEEE Trans. Image Process. **26**(7), 3196–3209 (2017)
10. Zhang, D., Meng, D., Li, C., Jiang, L., Zhao, Q., Han, J.: Co-saliency detection via a self-paced multiple-instance learning framework. IEEE Trans. Pattern Anal. Mach. Intell. **39**(5), 865–878 (2017)
11. Jian, M., Qi, Q., Dong, J., Sun, X., Sun, Y., Lam, K.: Saliency detection using quatemionic distance based weber descriptor and object cues. In: Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, pp. 1–4 (2016)
12. Jian, M., Lam, K., Dong, J., Shen, L.: Visual-patch-attention-aware saliency detection. IEEE Trans. Cybern. **45**(8), 1575–1586 (2015)
13. Han, J., Zhang, D., Cheng, G., Liu, N., Xu, D.: Advanced deep-learning techniques for salient and category-specific object detection: a survey. IEEE Signal Process. Mag. **35**(1), 84–100 (2018)
14. Shen, J., Peng, J., Dong, X., Shao, L., Porikli, F.: Higher-order energies for image segmentation. IEEE Trans. Image Process. **26**(10), 4911–4922 (2017)
15. Wang, W., Shen, J., Yang, R., Porikli, F.: Saliency-aware video object segmentation. IEEE Trans. Pattern Anal Mach Intell **40**(1), 20–33 (2018)
16. Han, J., Quan, R., Zhang, D., Nie, F.: Robust object co-segmentation using background prior. IEEE Trans. Image Process. **27**(4), 1639–1651 (2017)
17. Shen, J., Du, Y., Wang, W., Li, X.: Lazy random walks for superpixel segmentation. IEEE Trans. Image Process. **23**(4), 1451–1462 (2014)
18. Shen, J., Hao, X., Liang, Z., Liu, Y., Wang, W., Shao, L.: Real-time superpixel segmentation by DBSCAN clustering algorithm. IEEE Trans. Image Process. **25**(12), 5933–5942 (2016)
19. Cheng, G., Yang, C., Yao, X., Guo, L., Han, J.: When deep learning meets metric learning: remote sensing image scene classification via learning discriminative CNNs. IEEE Trans. Geosci. Remote Sens. **56**(5), 2811–2821 (2018)
20. Wang, K., Babenko, B., Belongie, S.: End-to-end scene text recognition. In: International Conference on Computer Vision, 1457–1464 (2011)
21. Wang, T., Wu, D., Coates, A., Ng, A.: End-to-end text recognition with convolutional neural networks. In: International Conference on Pattern Recognition, pp. 3304–3308 (2012)
22. Bissacco, A., Cummins, M., Netzer, Y., Neven, H.: PhotoOCR: Reading text in uncontrolled conditions. In: International Conference on Computer Vision, pp. 785-792 (2013)
23. Neumann, L., Matas, J.: Scene text localization and recognition with oriented stroke detection. In: International Conference on Computer Vision, pp. 97–104 (2013)
24. Lee, C., Bhardwaj, A., Di, W., Jagadeesh, V., Piramuthu, R.: Region-based discriminative feature pooling for scene text recognition. In: International Conference on Computer Vision and Pattern Recognition, pp. 4050–4057 (2014)
25. Yao, C., Bai, X., Shi, B., Liu, W.: Strokelets: a learned multi-scale representation for scene text recognition. In: International Conference on Computer Vision and Pattern Recognition, pp. 4042–4049 (2014)
26. Almazn, J., Gordo, A., Forns, A., Valveny, E.: Word spotting and recognition with embedded attributes. IEEE Trans. Pattern Anal. Mach. Intell. **36**(12), 2552–2566 (2014)
27. Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A.: Deep structured output learning for unconstrained text recognition. Eprint Arxiv. **24**(6), 603–611 (2015)
28. Su, B., Lu, S.: Accurate scene text recognition based on recurrent neural network. In: Asian Conference on Computer Vision, pp. 35–48 (2015)
29. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: International Conference on Computer Vision and Pattern Recognition, pp. 886–893 (2005)
30. Shi, B., Wang, X., Lyu, P., Yao, C., Bai, X.: Robust scene text recognition with automatic rectification. In: International Conference on Computer Vision and Pattern Recognition, pp. 4168–4176 (2016)
31. Wang, W., Shen, J., Shao, L.: Video salient object detection via fully convolutional networks. IEEE Trans. Image Process. **27**(1), 38–49 (2018)
32. Wang, W., Shen, J.: Deep visual attention prediction. IEEE Trans. Image Process. **27**(5), 2368–2378 (2018)
33. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning, pp. 4480–456 (2015)
34. Belagiannis, V., Wang, X., Shitrit, H., et al.: Parsing human skeletons in an operating room. Mach. Vis. Appl. **27**(7), 1035–1046 (2016)
35. Sebastien, R., Fredericm, J.: A novel target detection algorithm combining foreground and background manifold-based models. Mach. Vis. Appl. **27**(3), 363–375 (2016)
36. He, P., Huang, W., Qiao, Y., Loy, C., Tang, X.: Reading scene text in deep convolutional sequences. In: AAAI Conference on Artificial Intelligence, pp. 3501–3508 (2016)
37. Bengio, Y., Simard, P., Frasconi, P.: Learning long term dependencies with gradient descent is difficult. IEEE Trans. Neural Netw. **5**(2), 157–166 (1994)
38. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
39. Gers, F., Schraudolph, N., Schmidhuber, J.: Learning precise timing with LSTM recurrent networks. J. Mach. Learn. Res. **3**(1), 115–143 (2003)
40. Graves, A., Mohamed, A., Hinton, G.: Speech recognition with deep recurrent neural networks. In: International Conference on Acoustics, Speech, and Signal Processing, pp. 6645–6649 (2013)
41. Graves, A., Fernandez, S., Gomez, F., Schmidhuber, J.: Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: International Conference on Machine Learning, pp. 369–376 (2006)
42. Zeiler, M.: ADADELTA: An Adaptive Learning Rate Method. arXiv (2012)
43. Rumelhart, D., Hinton, G., Williams, R.: Learning internal representations by error propagation. Parallel Distrib. Process. **1**, 318–362 (1986)
44. Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A.: Synthetic data and artificial neural networks for natural scene text recognition. NIPS Deep Learning Workshop (2014)
45. Lucas, S., Panaretos, A., Sosa, L., Tang, A., Wong, S., Yang, R., et al.: Robust reading competitions: entries, results, and future directions. Int. J. Doc. Anal. Recognit. **7**(2), 105–122 (2005)
46. Mishra, A., Alahari, K., Jawahar, C.: Scene Text Recognition using higher OrScene text recognition using higher order language priors. In: British Machine Vision Conference, pp. 1–11 (2012)
47. Karatzas, D., Shafait, F., Uchida, S., Iwamura, M., Bigorda, L., Mestre, S., et al.: ICDAR 2013 robust reading competition. Int. Conf. Doc. Anal. Recognit. **2013**, 1484–1493 (2013)

48. Bhunia, A., Kumar, G., Roy, P., Balasubramanian, R., Pal, U.: Text recognition in scene image and video frame using color channel selection. Multimedia Tools Appl. **77**(7), 8551–8578 (2018)
49. Lee C., Osindero, S.: Recursive recurrent nets with attention modeling for OCR in the wild. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2231–2239 (2016)

Zhengchao Lei is currently pursuing the Ph.D. degree with the School of Computer Science, Beijing Institute of Technology, Beijing, China. His research interests include deep learning and pattern recognition in computer vision.

Sanyuan Zhao received her Ph.D. from Beijing Institute of Technology in 2012. She works at School of Computer Science and Technology of Beijing Institute of Technology as a lecturer since 2012. Her research areas include computer vision, deep learning, and virtual reality.

Hongmei Song is currently pursuing the master's degree at the School of Computer Science, Beijing Institute of Technology, Beijing, China. Her research interests include saliency detection and pattern recognition using deep learning in computer vision.

Jianbing Shen is currently a Professor with the School of Computer Science, Beijing Institute of Technology, Beijing, China. He received his Ph.D. from State Key Laboratory of CAD & CG of Zhejiang University, China. He has been a visiting scholar in Purdue University, ETH Zurich, National University of Singapore, Nanyang Technology University, and Chinese University of Hong Kong. He has obtained many flagship honors, including the Fok Ying Tung Education Foundation from the Ministry of Education, the Program for Beijing Excellent Youth Talents from the Beijing Municipal Education Commission, and the Program for New Century Excellent Talents from the Ministry of Education. His research interests include computer vision and deep learning.