Radical Composition Network for Chinese Character Generation

Mobai Xue[1], Jun Du[1,2(✉)], Jianshu Zhang[1], Zi-Rui Wang[1,3], Bin Wang[4], and Bo Ren[4]

[1] University of Science and Technology of China, Hefei, China
{xmb15,xysszjs,cs211}@mail.ustc.edu.cn, jundu@ustc.edu.cn
[2] Guangdong Artificial Intelligence and Digital Economy Laboratory (Pazhou Lab), Guangzhou, China
[3] Chongqing University of Posts and Telecommunications, Chongqing, China
wangzr@cqupt.edu.cn
[4] Youtu Lab, Tencent, Shenzhen, China
{bingolwang,timren}@tencent.com

Abstract. Recently, the generation of Chinese characters attracts many researchers. Many excellent works only focus on Chinese font transformation, which is, Chinese character can be transformed into another font style. However, there is no research to generate a Chinese character of new category, which is an interesting and important topic. This paper introduces a radical combination network, called RCN, to generate new Chinese character categories by integrating radicals according to the caption which describes the radicals and spatial relationship between them. The proposed RCN first splits the caption into pieces. A self-recurrent network is employed as an encoder, aiming at integrating these caption pieces and pictures of radicals into a vector. Then a vector which represents font/writing style is connected with the output from encoder. Finally a decoder, based on deconvolution network, using the vector to synthesize the picture of a Chinese character. The key idea of the proposed approach is to treat a Chinese character as a composition of radicals rather than a single character class, which makes the machine play the role of Cangjie who invents Chinese characters in ancient legend. As a kind of important resource, the generated characters can be reused in recognition tasks.

Keywords: Radical analysis · Chinese character generation · Data augmentation

1 Introduction

Chinese character generation is a challenging problem due to the large number of existing Chinese characters which is still increasing. Meanwhile, significant differences are observed between countless font/writing styles. The generation of chinese characters with novel classes and font/writing styles is a complex task.

J. Lladós et al. (Eds.): ICDAR 2021, LNCS 12821, pp. 252–267, 2021.
https://doi.org/10.1007/978-3-030-86549-8_17

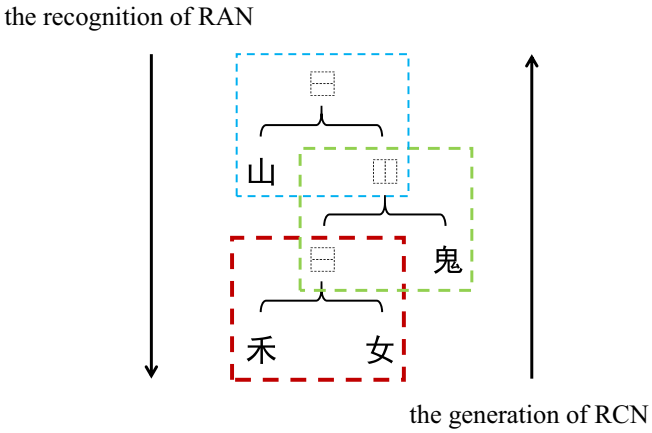

Fig. 1. The radical analysis by RAN and radical combination by RCN for the same character. The red, blue and green rectangles in RAN represent subtrees of recognition output. And in RCN, they represent the inputs of each step. (Color figure online)

In recent years, some teams investigate the font conversion of existing Chinese characters and have achieved success. [2] transfers the style of given glyph to the contents of unseen ones, capturing highly stylize fonts found in the real-world such as those on movie posters or infographics. [5] realizes the artistic style transfer of pictures and it can be used on the conversion of font style. [20] has achieved excellent results from font style transfer conditional adversarial networks based on encoder-decoder model. However, these algorithms can only generate Chinese characters with existent classes and have no ability to generated Chinese characters with novel classes. Moreover, these algorithms treat each Chinese character as a whole without considering the internal structure and basic components among Chinese characters.

In this paper, we propose a novel Chinese character generation network, called radical combination network (RCN), for generating characters of new categories. To create a new Chinese character, choosing a coding method that can be understood by computer is necessary. A number, a vector or a picture, neither of them is a good choice. This coding method needs to describe the internal details of Chinese characters, so that a new character can also be encoded in this way. A neural network named RAN [24] proposed a radical-based method to describe Chinese character. It treats a character as a composition of basic structural components, called radicals. The recognition results of RAN include radicals and spatial structures between them, which is a suitable encoding method for character generation. And only about 500 radicals [13] is adequate to describe more than 20,000 Chinese characters, which could reduce the cost of the project. Therefore, we design the RCN network according to the captions provided by RAN. RCN integrates radicals one by one at each step and finally outputs a picture of a Chinese character. Figure 1 shows the radical analysis by RAN and

radical combination by RCN for the same character. RAN treats a Chinese character as a tree, in which each leaf node represents a radical and each parent node represents the spatial structure of the two children. RAN recognizes a character following the tree in Fig. 1 from top to bottom, and RCN combines the radicals from bottom to top to realize the generation of a character. At step, radicals or the output from previous are integrated together and finally output a complete character.

It is clear to see that RCN attempts to imitate the development of Chinese characters. Chinese characters developed from simple ones to complex ones. Single-component characters were created first, then two-component characters were made up by two radicals, and then multi-component radicals appeared. People added a radical on an existing Chinese character to create new characters, according to the meaning, pronunciation or shape of radicals. RCN is designed to follow this characteristic.

Regarding to the network architecture, the proposed RCN is based on encoder-decoder model. A densely connected convolutional networks (DenseNet) [7,8,11,22] is employed to pre-process the pictures of radicals. We employ the network in work [8] and modify the parameters, so that the network can achieve better results in our tasks. We propose a tree-structured recurrent neural network (TRN) as the encoder to integrate the radicals. Refer to generative adversarial nets (GAN) [6,14,16], the decoder consists of several deconvolution networks [4,15]. Due to different mission objectives from GAN, we abandoned the discriminator and retained the generator.

As the new characters can be created, RCN provides a novel method on data augmentation. Although these generation samples may be blurry and unsightly for human vision, they can provide a great deal of information to recognition network.

The main contributions of this study are summarized as follows:

- We propose RCN for the Chinese character generation which is not dependent image input and realizes the transfer from caption to image.
- Based on radicals, the proposed method can create new Chinese characters. RCN can generated Chinese characters with novel classes and font/writing styles by the novel captions and mixing up font vectors.
- We experimentally demonstrate how the generated data improve the recognition rate, especially for unseen characters in printed/handwritten.
- RCN can increase the diversity of data sets and reduce the cost of collection and annotation effectively.

2 Related Works

2.1 Recursive Neural Network

In the past few years, lots of efforts have been made for recursive neural network. Tree-LSTM [19] proposes a generalization of LSTMs to tree-structured network topologies. Improved tree-LSTM networks [1,25] propose effective mothods for

Natural Language Processing. Recursive neural networks [12,18] perform well on sentiment classification at the sentence level and phrase level. Refer to the above works, we modify the framework to adapt to Chinese character generation. The proposed encoder TRN is an improved recurrent neural network and designed to matching the input caption, which can receive three input vectors.

2.2 Data Augmentation

Traditional methods of data augmentation adopt image processing on training set upon the retaining the captions. The easiest and most common method of data augmentation label-preserving transformations [10], including translation, horizontal mapping and changing the value of RGB channels. SMOTE [3] uses random interpolation on training set to synthesis data. Mixup [23] adopts linear interpolation to mix up two samples as a new sample. Compared with traditional methods, RCN can create samples with new captions, which greatly improve the diversity.

3 Radical Analysis and Caption Segmentation

RAN proposes a method to captioning Chinese characters based on detailed analysis of Chinese radicals and structures. It decomposes Chinese characters into corresponding captions, and we employ the captions to generate Chinese characters. As for spatial structures among radicals, [24] shows ten common structures in Fig. 2, where "a" represents a left-right structure, "d" represents a top-bottom structure, "stl" represents a top-left-surround structure, "str" represents a top-right-surround structure, "sbl" represents a bottom-left-surround structure, "sl" represents a left-surround structure, "sb" represents a bottom-surround structure, "st" represents a top-surround structure, "s" represents a surround structure and "w" represents a within structure. They use a pair of braces to constrain a single structure in character captions. In the existing Chinese characters, these ten structures are not uniform distribution. "a" and "d" appear more often.

Before these captions are sent into RCN, the complete captions need to be split to pieces, as shown in the Fig. 2. Each piece includes 3 symbols and the first is a spatial symbol which describes the spatial structure between the other two symbols. The second and the third symbol could be radical characters or filled character "#". A radical character represents an existing radical and filled character "#" represents a subtree of the caption. These caption pieces describe parts of characters, which are the inputs of RCN.

4 Network Architecture of RCN

RCN based on encoder-decoder model is shown in Fig. 3 and it is a reverse form of RAN. First, the caption of the character is clustered into several pieces

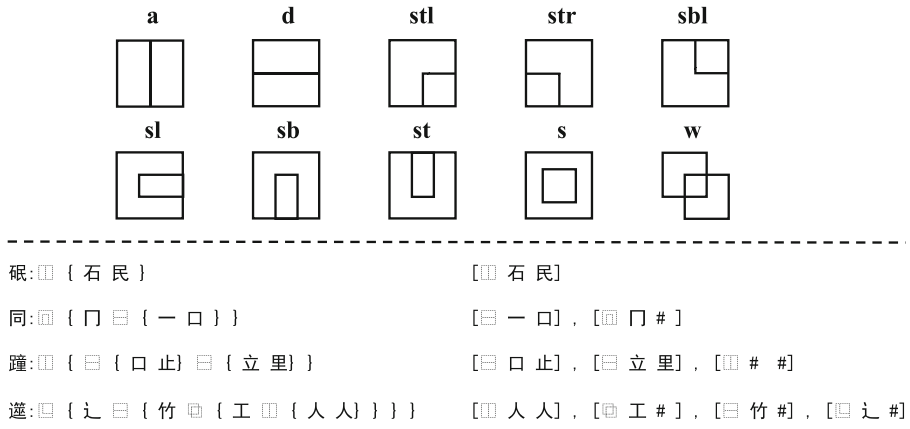

Fig. 2. Graphical representation of ten common spatial structures between Chinese radicals and several examples of captions from sequence to pieces.

according to structural relationship and the encoder of RCN chronologically fuses them into high-level representations with a self-recurrent mechanism. Then, a vector which represents font/writing style is connected with the output from encoder. Finally, the decoder of RCN with deconvolution neural network utilizes the representation at the end to generate a Chinese character picture.

4.1 Data Processing

The description of a captation can form a tree perfectly. Naturally, a caption is expanded into piece sequence $\boldsymbol{C}$ according to tree depth-first traversal order, which is described in Sect. 3. Each piece $\boldsymbol{c}_i \in C$ includes 3 basic components c_{i0}, c_{i1} and c_{i2}.

$$\boldsymbol{C} = \{\boldsymbol{c}_1, \boldsymbol{c}_2, ..., \boldsymbol{c}_T\} \ , \ \boldsymbol{c}_i \in \mathbb{R}^3 \tag{1}$$

where T is the length of $\boldsymbol{C}$.

We employ three learnable codebooks $(\boldsymbol{G_s}, \boldsymbol{G_r}, \boldsymbol{G_f})$ to represent the spatial structure relationship, radical skeleton and font/writing style.

4.2 TRN Encoder

Encoder is a tree-structured recurrent neural network, called TRN. TRN consists of basic units, TRN cell, which is multiply used in the process of generation. In this section, we focus on the architecture of the encoder: generation process of TRN and the self-recurrent structure of TRN and the details of TRN cell.

Overall Architecture of TRN. The structure of TRN is not fixed, which depends on the specific caption. So TRN can be applied to tree-structured caption with variable depth. Figure 4 shows an example of encoding captions, which

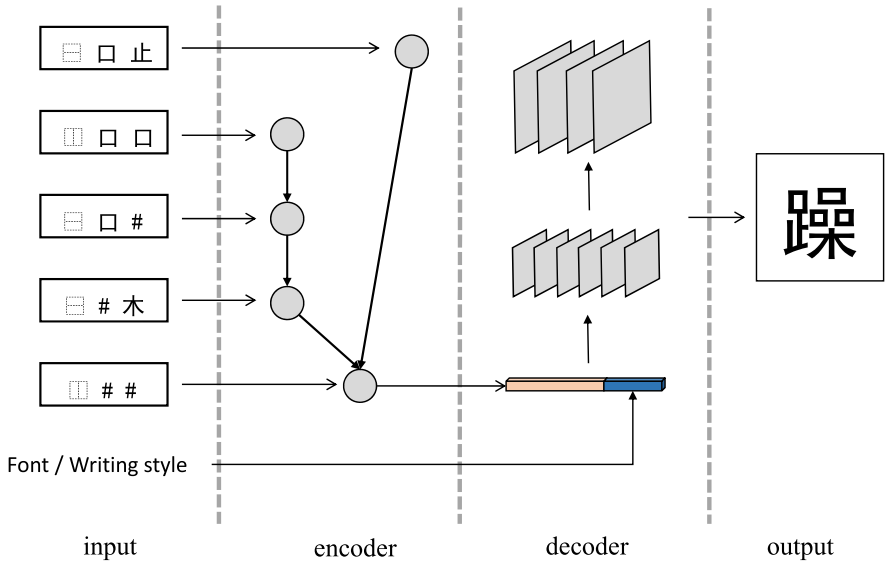

Fig. 3. The overall architecture of RCN for Chinese character generation. We employ a tree-structured encoder and each circle represents an encoder cell.

is shown in two different ways: the tree-structured following the logical structure and the sequence-structure following the timing sequence.

The input caption is expanded into piece sequence $\boldsymbol{C}$ according to tree depth-first traversal order, which is described in Sect. 3. Each piece $\boldsymbol{c}_i \in C$ includes 3 basic components c_{i0}, c_{i1} and c_{i2}. At each step, TRN cell generates a vector according to the local description of the caption c_i. As shown in Figs. 4 the tree-structured shows the structure of a character intuitively, which is consist with human cognition. However, the sequence-structure is used in code implementation. The input caption is treated as a sequence following the depth-first traversal order of the tree.

Tree-Structured Recurrent Network of TRN. In this section, we introduce how the tree-structured recurrent of TRN is implemented. Compared with traditional sequence recurrent network, tree-structured recurrent network calls the output across several steps, not limited to the previous step. In actual processing, we need memory module (a LIFO stack) $\boldsymbol{S}$ to store the outputs $\boldsymbol{O}_t$ from each step t.

At each step t, we input the local description of the caption $\boldsymbol{c}_i$ into TRN cell and get the output $\boldsymbol{O}_t$. Send $\boldsymbol{O}_t$ into the memory module $\boldsymbol{S}$ at step t, and take $\boldsymbol{O}_t$ out when it is called in further step. The memory module ensures the output $\boldsymbol{O}_t$ can be called correctly in subsequent operations. At step T (the length of

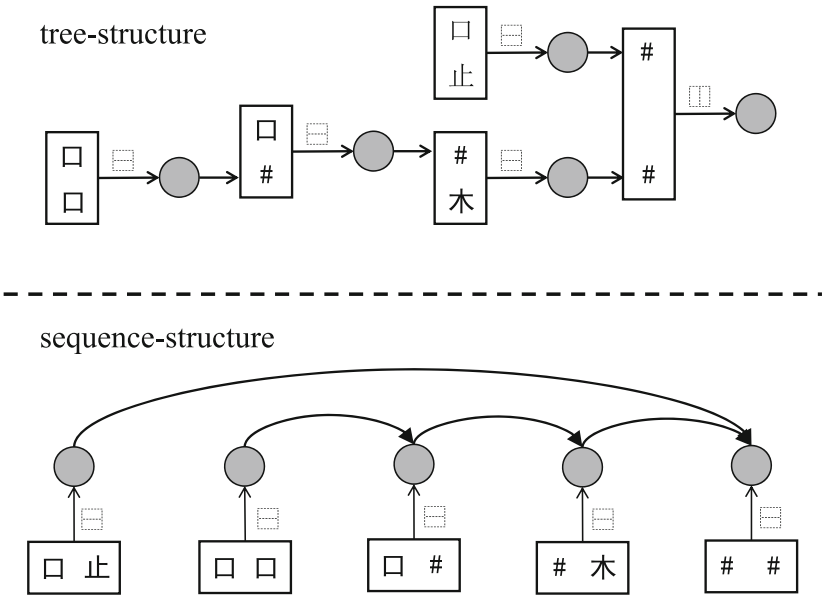

Fig. 4. The architecture of encoder TRN and each circle represents a TRN cell. The tree-structure follows the logical structure and the sequence-structure follows the timing sequence.

$\boldsymbol{C}$), the caption is completely encoded. There must be only one item ($\boldsymbol{O}_T$) in the memory module $\boldsymbol{S}$ and the output is sent to decoder to generate a character.

It is easier to understand TRN by analogy with RNN, whose extended structure is sequence. TRN and RNN both are recurrent neural network, and the outputs are both obtained by iteration. But they update the intermediate results in different ways. At each time step t, the hidden state $\boldsymbol{h}_t$ of the RNN is updated by

$$\boldsymbol{h}_t = f_{\mathrm{RNN}}(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}) \tag{2}$$

where f_{RNN} is a non-linear activation function and $\boldsymbol{x}_t$ is the input vector. For TRN, the interim output $\boldsymbol{O}_t$ is updated by Eq. 3.

$$\boldsymbol{O}_t = f_{\mathrm{TRN}}(\boldsymbol{c}_t, S) \tag{3}$$

f_{RNN} only calls $\boldsymbol{h}_{t-1}$ at step t, while f_{TRN} may call $\boldsymbol{O}_{t-n}$ where $0 < n < t$. The two child nodes of the parent node may all be subtrees, which causes the difference between tree-structured and sequence-structure. As shown in Fig. 4, at the final step, f_{TRN} calls the output from step 3 and step 0.

TRN Cell. The processing of TRN encoding is realized by running TRN cell several times. TRN cell processes input caption $\boldsymbol{c}_t$ into high dimensional representation. Figure 5 shows the architecture of TRN cell.

At each step t, input caption piece $\boldsymbol{c}_t$ includes c_{t0}, c_{t1} and c_{t2}. They are transformed into high-dimensional representations through codebooks $\boldsymbol{G}_s$ and $\boldsymbol{G}_r$, which represent the spatial relationship and radical skeleton. We acquire the quantized representation of spatial character c_{t0} from codebook $\boldsymbol{G}_s$ and c_{t0} is represented as a l-dimensional vector $\boldsymbol{e}_{t0}$. Since c_{t1} and c_{t2} could be radical characters or filled characters, we need to judge how the corresponding vectors $\boldsymbol{e_{t1}}$ and $\boldsymbol{e_{t2}}$ is obtained.

$$\begin{aligned}
\boldsymbol{e}_{t2} &= \begin{cases} \boldsymbol{G}_s[c_{t2}] & c_{t2} \text{ is radical character} \\ \boldsymbol{s}_{-1} & c_{t2} \text{ is filled character} \end{cases} \\
\boldsymbol{e}_{t1} &= \begin{cases} \boldsymbol{G}_s[c_{t1}] & c_{t1} \text{ is radical character} \\ \boldsymbol{s}_{-1} & c_{t1} \text{ is filled character } \& \, c_{t2} \text{ is radical character} \\ \boldsymbol{s}_{-2} & c_{t1} \text{ and } c_{t2} \text{ are both filled character} \end{cases}
\end{aligned} \tag{4}$$

where $\boldsymbol{G}_s[c_{t1}], bmG_s[c_{t2}], \boldsymbol{e}_{t1}, \boldsymbol{e}_{t2} \in \mathbb{R}^L$. $\boldsymbol{s}_{-1}$ and $\boldsymbol{s}_{-2}$ represent the last and the second to last item in memory module $\boldsymbol{S}$.

Then, concate $\boldsymbol{e}_{t0}$, $\boldsymbol{e}_{t1}$ and $\boldsymbol{e}_{t2}$.

$$\boldsymbol{e}_t = \{\boldsymbol{e}_{t0}, \boldsymbol{e}_{t1}, \boldsymbol{e}_{t2}\} \tag{5}$$

where $\boldsymbol{e}_t \in \mathbb{R}^{2L+l}$. We use a feed forward network to change the dimension to L, which ensures that the format of output $\boldsymbol{O}_t$ is match when it is called. The feed forward network consists of 3 linear transformations with 2 ReLU activation in between.

$$\begin{aligned}
\boldsymbol{O}_t &= \mathrm{FFN}_0(\boldsymbol{e}_t) \\
&= \max(\max(0, \boldsymbol{e}_t W_0 + b_0) W_1 + b_1) W_2 + b_2
\end{aligned} \tag{6}$$

where $W_0 \in \mathbb{R}^{2L+l \times 2L}, W_1 \in \mathbb{R}^{2L \times L}, W_2 \in \mathbb{R}^{L \times L}$.

4.3 Decoder

The decoder aims to generate a picture of the target Chinese character, which is a grayscale image. First, the font/writing style is transformed into high-dimensional representations through codebooks $\boldsymbol{G}_f$ and $\boldsymbol{O}^*$ is the concatenation of the font/writing style vector and the output $\boldsymbol{O}_T$ from encoder. A feed forward network is used to enhance the fitting ability of the decoder and further fuse the skeleton information and font style. The feed forward network consists of 2 linear transformations with a ReLU activation in between.

$$\mathrm{FFN}_1(\boldsymbol{O}^*) = \max(0, \boldsymbol{e}_t W_3 + b_3) W_4 + b_4 \tag{7}$$

where $W_3 \in \mathbb{R}^{L \times 2L}, W_4 \in \mathbb{R}^{2L \times L}$. Then, the decoder consists of four deconvolution layers [4] and decodes $\boldsymbol{O}^*$ into an array $\hat{\boldsymbol{Y}}$, where $\boldsymbol{O}^* \in \mathbb{R}^{L \times 1 \times 1}$ and $\hat{\boldsymbol{Y}} \in \mathbb{R}^{1 \times H \times W}$. During decoding, the number of channels is reduced from L to 1,

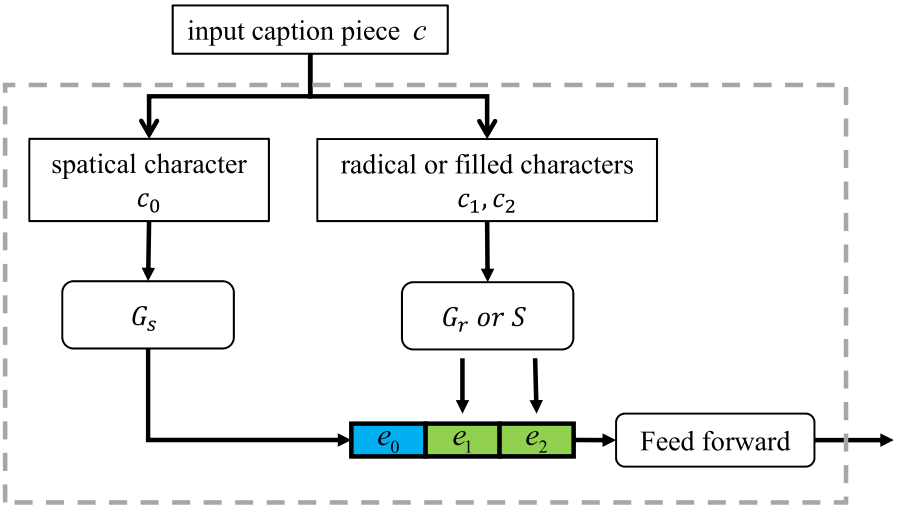

Fig. 5. The architecture of TRN cell. The spatial characters and radical characters are transformed into high-dimensional representations through codebooks. The filled character represents the item in memory module $\boldsymbol{S}$

and the size is expanded from 1×1 to $H \times W$. Finally, we use contrast stretching operation for $\hat{\boldsymbol{Y}}$ and achieve normalization.

$$\bar{\boldsymbol{Y}} = \frac{\hat{\boldsymbol{Y}} - \min(\hat{\boldsymbol{Y}})}{\max(\hat{\boldsymbol{Y}}) - \min(\hat{\boldsymbol{Y}})} \tag{8}$$

RCN can generate novel font/writing style by weighted addition on font/writing style vectors and a novel caption can be transformed into a novel Chinese character. These new samples can increase the diversity of data sets and reduce the cost of collection and annotation effectively.

4.4 Perceptual Loss

We employ perceptual loss function [9] that measure high-level perceptual and semantic differences between generated sample and ground truth. Rather than encouraging the pixels of the generated image to exactly match the pixels of the target image, we instead encourage them to have similar feature representations as computed by the loss network φ. Let $\varphi_j(x)$ be the activations of the jth layer of the network φ when processing the image; if jth layer is a convolutional layer then $\varphi_j(x)$ will be a feature map of shape $C_j \times H_j \times W_j$. The feature reconstruction loss is the (squared, normalized) Euclidean distance between feature representations:

$$l_j = \frac{\left\|\varphi_j(\boldsymbol{Y}) - \varphi_j(\bar{\boldsymbol{Y}})\right\|}{C_j H_j W_j} \tag{9}$$

5 Experiment

The innovation of our work is that Chinese characters are generated by using the defined radicals. The generated Chinese characters should be similar to printed style or handwritten style. Therefore, RCN creates new Chinese characters and also provides a novel method for data augmentation. In this section, we introduce several experiments to show the effect of RCN. Section 5.1 explains the network parameters and describes the division of database. Section 5.2 shows some generated samples of RCN and analyses the restrictions on random captions. Section 5.3 illustrates how generated samples can improve the recognition rate of RAN on unseen characters. In Sect. 5.4, we verify the effectiveness of this method on handwritten characters and show the improvement of the recognition.

5.1 Experimental Parameters and Database

In TRN cell, dimension l of vectors in codebook G_s is set to 256 and dimension L of vectors in codebook G_r is set to 1024. The decoder consists of four deconvolution layers of kernel size 4×4 with stride 2 and padding 1. The dimension of vectors in codebook G_f is set to 1024. The dimensions of output vectors from each layer are 256, 64, 8 and 1, respectively. In all our experiments, the loss network φ is the 16-layer VGG network [17] and the pre-trained model is directly acquired from torchvision package.

As for the recognition model, RAN has the same configuration as [21]. A DenseNet is employed as the encoder. The initially convolutional layer comprises 48 convolutions of size 7×7 with stride 2 and each block comprises 10 bottleneck layers and growth rate is set to 24. The decoder is two unidirectional layers with 256 GRU units.

We prepare the printed and handwritten data sets and design related experiments to verify the effectiveness of our method in simple and complex scenes. In order to generate new Chinese characters, we prepare 20,000 random captions of nonexistent Chinese characters. We introduce the details and partitions of the data in the following.

Printed Database. In the experiments on printed characters, we choose 27,533 Chinese characters in 5 font styles (Hei, Song, Kai, Deng, Fangsong) as database and divide them into training set and testing. There are 114,665 printed characters in 5 font styles in training set, which contains 22,933 classes. And there are 20,860 printed characters in testing set, which contains 4,172 classes that do not appear in training set.

Handwritten Database. The handwritten database is the CASIA database including HWDB1.0, 1.1 and 1.2. We use HWDB1.0 and 1.1 set as training set which contains 3,755 classes, 720 writing styles and 2,674,784 handwritten

Chinese characters. HWDB1.2 set is employed as testing set and we choose 3,277 unseen classes for evaluating the performance on unseen Chinese characters. We make sure all radicals of these characters are covered in 3,755 training classes. Note that the Chinese characters in HWDB1.2 set are not common and usually have more complicated radical composition.

Random Caption. The network generates the specified characters according to the description of the input caption. It is feasible to generate novel/nonexistent Chinese characters by inputting corresponding captions. However, the caption cannot be completely random due to the limitation of the default composition rules of Chinese character structure. A radical has given positions in character, which is determined by the shape of the radical. For example, it's unusual to observe a radical with vertical bar shape appears in a top-bottom structure.

Therefore, we need to count the frequency of the combination of each radical and spatial structure. It is necessary to avoid the combination that does not appear in the training set when generating random captions. We generate a random caption from top to bottom according to the tree structure. First, we select a spatial structure as the parent node randomly. Then, the radicals that appear in such structure are selected as the corresponding child nodes. In order to generate tree structure annotated with different depths, we replace radical by using a subtree with probability 0.2 and continue to select child nodes until the tree structure is complete.

5.2 Experiments on Generation of RCN

In this section, we show some generated results of RCN. In the experiments on printed characters, we generate new Chinese characters with different font styles, and verify the influence of training set capacity on the generation effect. In the experiments on handwritten characters, we expand the categories and writing style of the database.

In the experiments on printed characters, we use the printed training set mentioned in Sect. 5.1 and increase the training set from 4,586 to 22,933 with 20% stride to observe the change of generation effect. The printed testing set is employed to validate the effect on unseen characters. As shown in the top half of the Fig. 6, the generated results get clearer and the details become smoother with the increasing of training set. We display some results of testing set in the middle part of Fig. 6 and some results of random caption input in the lower part.

In the experiments on handwritten characters, we use the HWDB training set and testing set mentioned in Sect. 5.1. As shown in the top half of the Fig. 7, we generate mixed-up writing style characters that are mentioned in Sect. 4.3. We display some characters with testing classes in the lower part.

Fig. 6. The generated result of printed Chinese characters. The first line shows 6 groups of generated results with the increasing training data, the obtained characters become more and more clearly. The middle part shows some results of testing set and the lower part shows some results of random caption inputs.

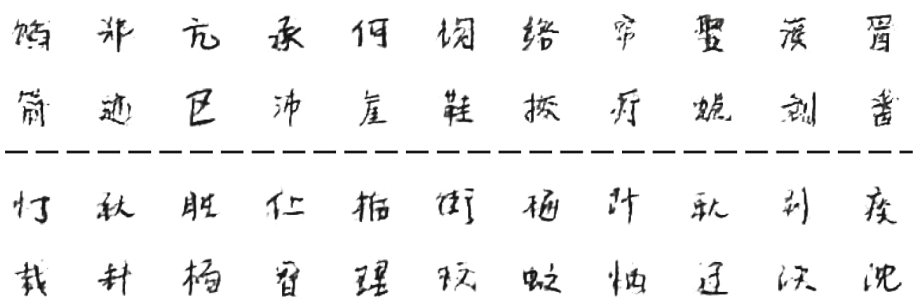

Fig. 7. The generated result of handwritten Chinese characters. The top half part shows mixed-up writing style characters, and the lower part shows the characters with testing classes.

In this experiment, we find that captions contain semantic information which can be regarded as sentences. We consider that semantic information in captions can not be intuitively understood. Based on the observation of the experimental results, the generated results created by testing captions perform much better than the ones created by random captions. It can be explained via analogy with a translation work. If a verb in a sentence is replaced by a noun, the sentence may be translated into an incorrect one. Despite the grammatical constraints, a sentence made up of random words may probably become an outlier for translation network. It is the same reason to RCN, an unsuitable radical in a caption may lead to a blurry output, although we create the captions according to the restriction of radicals' shape. However, the characters in the testing set are

Table 1. Result of experiment on printed characters.

Training data			Accuracy					
Training	Testing	Random	Hei	Song	FangSong	Deng	Kai	Avg
114,665	-	-	88.68	88.56	87.19	87.17	86.80	87.79
114,665	20,860	-	92.97	92.84	91.91	91.89	90.00	92.03
114,665	20,860	20,000	93.81	93.93	92.94	92.78	90.11	92.83
114,665	20,860	40,000	95.16	95.04	94.01	93.95	91.23	93.99
114,665	20,860	80,000	95.00	94.90	93.99	94.20	92.46	**94.22**

existing ones and the captions contain correct semantic information. These captions are fitter to the training set and radicals in the generated results are clearer and more distinguishable.

We also observe that the performance of generated results can not be completely random. In the one hand, captions are limited by the default rules of existent Chinese character structure, which is mentioned in Sect. 4.3. On the other hand, a radical may have various performances, which depends on its position in character. A radical may perform differently in different positions of a character. It may be a smaller one at the left in a left-right structure, which is learned from existing Chinese characters in the training set. These constraints make generated results fit to the human cognition.

5.3 Data Augmentation Experiments on Printed Characters

In this section, we use the newly created classes as data augmentation and show how the generated results improve the recognition rate on unseen characters. In this experiment, we use the printed training set to train the recognition model RAN and observe the increase of recognition rate by adding new samples. We add 20,860 generated samples with testing class and 80,000 with random classes in 5 font styles. The printed testing set is used for evaluating the performance on unseen characters.

The recognition rate of each font and the whole test set are shown in Table 1. The part of training data displays the number of samples. 'Training' represents the printed training set; 'Testing' represents the generated samples with testing classes and 'Random' represents the generated samples with random classes. With the addition of the generated samples of testing classes, the accuracy increases from 87.79% to 92.03%. And the accuracy increases to 94.22% further with the generated samples of random classes increase from 20,000 to 80,000.

5.4 Data Augmentation Experiments on Handwritten Characters

In this section, we use the newly created classes and writing styles to train the recognition model, and show the improvement of recognition rate on characters with unseen classes and writing style. In this experiment, we use the handwritten

Table 2. Result of experiment on handwritten characters.

Training data				Accuracy
Raw	Classes	Styles	Classes+Styles	
2,674,784	-	-	-	40.82 [21]
2,674,784	1,440,000	-	-	43.23
2,674,784	2,880,000	-	-	44.17
2,674,784	5,760,000	-	-	44.89
2,674,784	-	1,126,500	-	42.50
2,674,784	-	2,253,000	-	43.22
2,674,784	-	4,506,000	-	43.66
2,674,784	5,760,000	4,506,000	-	45.42
2,674,784	5,760,000	4,506,000	9,600,000	**46.12**

training set to train the recognition model RAN and the handwritten testing set is used for evaluation. We use the generated samples to train the network and observe the impact of novel classes and writing styles on the recognition effect.

We reproduce the experimental result in [21] as a baseline. As shown in Table 2, we expand the training data on classes and writing styles. 'Raw' represents the handwritten training set mentioned in Sect. 5.1; 'Classes' represents the generated samples with novel classes that are not shown in the training set; 'Styles' represents the samples with mixed-up writing styles and 'Classes+Styles' represents the samples with novel classes and writing styles at the same time. We observe that the newly created classes are more helpful in recognizing characters with unseen classes and unseen writing styles. When all the generated samples are used to train the recognition model, the recognition rate increases from 40.82% to 46.12%.

6 Conclusion

In this paper, we introduce a novel model named radical combination network for Chinese character generation. We show the function of the generation for Chinese characters with novel classes and font/writing styles by RCN, and the effect of data augmentation on recognizing unseen Chinese characters. In future work, we plan to improve the quality of generated images and apply this work to the generation of natural scene text with complex background. We also will employ dual learning and jointly optimize the generation and recognition tasks for better performance.

Acknowledgements. This work was supported in part by the MOE-Microsoft Key Laboratory of USTC, and Youtu Lab of Tencent.

References

1. Ahmed, M., Samee, M.R., Mercer, R.E.: Improving tree-LSTM with tree attention. In: 2019 IEEE 13th International Conference on Semantic Computing (ICSC), pp. 247–254. IEEE (2019)
2. Azadi, S., Fisher, M., Kim, V.G., Wang, Z., Shechtman, E., Darrell, T.: Multi-content gan for few-shot font style transfer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7564–7573 (2018)
3. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: synthetic minority over-sampling technique. J. Artif. Intell. Res. **16**, 321–357 (2002)
4. Cho, S., Wang, J., Lee, S.: Handling outliers in non-blind image deconvolution. In: 2011 International Conference on Computer Vision, pp. 495–502. IEEE (2011)
5. Gatys, L.A., Ecker, A.S., Bethge, M.: Image style transfer using convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2414–2423 (2016)
6. Goodfellow, I., et al.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, pp. 2672–2680 (2014)
7. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
8. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708 (2017)
9. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9906, pp. 694–711. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46475-6_43
10. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
11. Larsson, G., Maire, M., Shakhnarovich, G.: FractalNet: ultra-deep neural networks without residuals. arXiv preprint arXiv:1605.07648 (2016)
12. Li, J., Luong, M., Jurafsky, D., Hovy, E.: When are tree structures necessary for deep learning of representations. Artificial Intelligence. arXiv (2015)
13. Li, X., Zhang, X.: The writing order of modern Chinese character components. J. Modernization Chin. Lang. Educ. **2**, 26–41 (2013)
14. Mirza, M., Osindero, S.: Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784 (2014)
15. Noh, H., Hong, S., Han, B.: Learning deconvolution network for semantic segmentation (2015)
16. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)
17. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. Computer Science (2014)
18. Socher, R., et al.: Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1631–1642 (2013)
19. Tai, K.S., Socher, R., Manning, C.D.: Improved semantic representations from tree-structured long short-term memory networks. Computation and Language. arXiv (2015)

20. Tian, Y.: zi2zi: Master Chinese calligraphy with conditional adversarial networks (2017)
21. Wang, W., Zhang, J., Du, J., Wang, Z.R., Zhu, Y.: DenseRAN for offline handwritten Chinese character recognition. In: 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR), pp. 104–109. IEEE (2018)
22. Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1492–1500 (2017)
23. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: beyond empirical risk minimization. arXiv preprint arXiv:1710.09412 (2017)
24. Zhang, J., Zhu, Y., Du, J., Dai, L.: Radical analysis network for zero-shot learning in printed Chinese character recognition. In: 2018 IEEE International Conference on Multimedia and Expo (ICME), pp. 1–6. IEEE (2018)
25. Zhu, X., Sobihani, P., Guo, H.: Long short-term memory over recursive structures. In: International Conference on Machine Learning, pp. 1604–1612. PMLR (2015)