Dual Learning of the Generator and Recognizer for Chinese Characters

Yixing Zhu, Jun Du and Jianshu Zhang
National Engineering Laboratory for Speech and Language Information Processing
University of Science and Technology of China
Hefei, Anhui, China
zyxsa@mail.ustc.edu.cn, jundu@ustc.edu.cn, xysszjs@mail.ustc.edu.cn

Abstract

Recently, deep learning based approaches become increasingly popular for both generation and recognition of Chinese characters. In this study, we propose a dual learning framework of offline Chinese character generator (G) and recognizer (R) based on deep models. The motivation is that G and R can well collaborate to improve the performance for both. On one hand, the learning of G, aiming at the font style transfer, is enhanced via the regression loss defined on the intermediate layers of R and the output layer of G rather than only the output layer of G. On the other hand, the learning of R is boosted by using the augmented data from the generator G to improve the recognition of character classes with unseen font style. Tested on the dataset of printed Chinese characters with a vocabulary of 3755, the proposed approach can significantly improve both the recognition performance of R and the generation performance of G with high-resolution details for the adaptation of the unseen font styles.

1. Introduction

Both Chinese character recognition and Chinese character generation have attracted considerable attention in recent decades. They have wide application in many areas, such as business cards, natural scene images recognition and Chinese font styles transfer. As a research problem, automatic recognition or generation of Chinese characters is different from these problem in English [2] and exhibits several fascinating challenges. For example, the complicated geometric structures and enormous categories make Chinese character recognition or generation difficult.

Recently, deep learning based Chinese character recognition and generation have made a significant breakthrough [27, 25]. However, there are still some problems. On one hand, when dealing with printed Chinese character recognition, the recognition accuracy on unseen font styles remains undesirable which results from incomplete coverage of font styles of printed Chinese characters in train set. It is a straightforward way to append diverse font styles into train set as many as possible, which will then bring enormous computational cost. On the other hand, the traditional neural network based character generator, which is learned via the regression loss defined on the output layer of generator [2], is hard to produce high-quality Chinese characters in some special font styles. Overall, in traditional optimization methods, Chinese character generator and Chinese character recognizer have no interaction between each other. They are optimized separately.

In this paper, we introduce a dual learning of the generator G and recognizer R for Chinese characters . Inherently unlike traditional methods [22, 15, 14] both the generator G and the recognizer R are convolutional neural networks (CNN), and we present a concept that the generator G and the recognizer R should help with each other during optimization. The proposed dual learning of G and R address the limitations that have been mentioned before. In generation step, the learning of G, aiming at the font style transfer, is enhanced via the regression loss defined on R's feature maps rather than only the output layer of G. As a result, the recognizer R become a guider of G and the generated Chinese characters are more readable than separated learning. Meanwhile, in recognition step, when recognizer R meets some Chinese characters with unseen font styles, the generator G will first imitate these unseen font styles and then generate them. We anneal the recognizer R with the newly generated Chinese character samples. Hence, the recognizer R will be more adaptable of the unseen font styles after the fine-tuning procedure. We validate the advantage of dual learning of the generator and the recognizer by compare it with models under separated learning on the dataset of printed Chinese characters with a vocabulary of 3755.

The contributions of this paper are as follows:

1. We introduce a novel dual learning method of Chinese character generation and Chinese character recognition. In our proposed method, the generator and recognizer are optimized with the help of each other.

2327-0985/17 $31.00 © 2017 IEEE
DOI 10.1109/ACPR.2017.73

2. We have shown through visualization that the generator G based on dual learning can produce more readable Chinese characters.

3. In experiments, we have shown that the recognition accuracy of recognizer R in unseen font styles is improved with the help of G.

2. Related Work

With the pleasant performance which neural network brings, there has been a great breakthrough in the field of Chinese character recognition. In [3], neural network was first applied in Chinese character recognition. Then [21] proposed a deep learning framework based on relaxation convolutional neural network (R-CNN) which is the winner of ICDAR 2013 Chinese Handwriting Character Recognition Competition. In the area of printed Chinese character recognition (PCCR) [27] classfied 3755 printed Chinese characters via multi-pooling alexnet [12] in 280 different fonts. Recently, some people also combined traditional feature extraction methods and deep learning based methods together (*e.g.* [28], [25]), [25] to achieve high accuracy for both online HCCR (Handwritten Chinese Character Recognition) and offline HCCR, and the memory cost of their offline HCCR system has been greatly decreased.

In the area of font style transfer, [2] learned law of 4 basic English alphabets in a large number of font to synthesize all the remaining alphabets and [4] presented a new discriminative linear regression approach for Chinese OCR. [26] proposed a framework by using RNN (Recurrent Neural Networks) to generate online handwritten character. [15] generates all missing Chinese characters via collecting subset from a small number of Chinese characters.

In recent years, GAN (Generative Adversarial Nets) [6] has been applied in many areas such as image processing, image generation and so on. [20] also made a font transfer system named zi2zi by employing GAN [6] and achieved very good result. However, most of them (*e.g.* [23], [13], [10]) only focus on the results of the generator and did not care about the recognizer. Consequently, inspired by GAN but different from it, we present a dual learning method where the generator and the recognizer will not compete with each other but help with each other. More specifically, we applies a recognizer R to recognize printed Chinese characters. There are 3755 frequently-used Chinese characters (level-1 set of GB2312-80). While the generator G is employed to generate printed Chinese characters of different font styles, and the generated samples are then fed into R. The optimization of G is also guided by R. Overall, by helping each other, the performance of the G and the R will be improved.

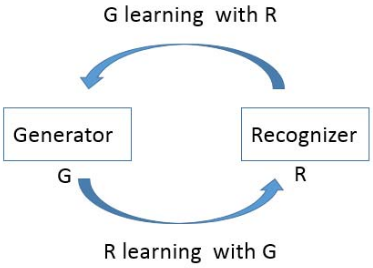

Figure 1. Our dual learning framework of offline Chinese character generator (G) and recognizer (R) based on deep models. the learning of G, aiming at the font style transfer, is enhanced via the regression loss defined on the intermediate layers of R and the output layer of G rather than only the output layer of G. Then R is boosted by using the augmented data from the generator G to improve the recognition in unseen font style.

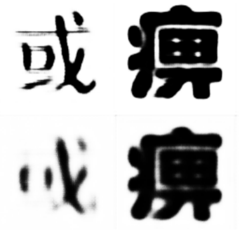

Figure 2. If we define loss on the output layer of G directly, then the generator can not determine which pixels of target font is important, the generated character is not readable. (the upper figure is generateed with recognizer's helps, the remaining two characters is directly generated by MAE loss between target and generated figure.)

3. The Dual Learning Framework

3.1. System Overview

In this paper, we propose a dual learning training method for Chinese character generator and Chinese character recognizer. Rather than optimized in their separately way, our generator and recognizer can help with each other during the optimization procedure. The overall dual learning procedure is shown in Figure 1. We first pre-train an inferior recognizer R by using a common supervised learning method. Afterwards, R begin to guide the learning of generator G. The generator G, which is aiming at the font style transfer, is optimized via the regression loss defined on the intermediate layers of R and the output layer of G rather than only the output layer of G. Because we think the recognizer can help the generator catch the flag feature of the target font. It is shown in Figure 2 that G, which is trained with the help of R, can generate more readable Chinese characters than separately trained G. We then boost R by using

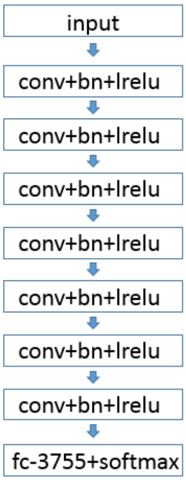

Figure 3. The sketch of our recognizer.

the augmented data produced by generator G to improve R's adaptation ability of the Chinese characters with unseen font styles. Therefore, the recognizer R and the generator G can be trained in a cooperated way.

3.2. Recognizer

The structure of R is shown in Figure 3. It is composed of an input layer, a stack of convolutional layers and a softmax layer. Each convolutional layer consists of a convolution filter, a batch normalization layer and a lrelu nonlinear activation layer (represented as conv + bn + lrelu).

Convolution filter has made a great breakthrough recently in the field of Chinese character recognition (*e.g.* [28], [25], [21], [7]). It is employed as an outstanding tool when dealing with images due to its highly invariant to translation, scaling, tilt or other types of image modifications. [24] successful used convolution neural networks in the handwritten mathematical expression recognition, and achieved state-of-the-art results. Moreover, conv filter does not require traditional handcrafted feature extraction and data reconstruction processes before recognizing an image [12]. The batch normalization layer is performed before the activation layer. We train R by using batch normalization to reduce the internal covariate shift. Hence, we do not need to concern about the initialization of parameters and the convergence process will be speeded up. The nonlinear activation function used here is lrelu (Leaky-Relu). Leakly-Relu has been proved to be more powerful than common Relu activation function recently (*e.g.* [18], [25]). It is defined as $f(x) = max(x,0) + \lambda min(x,0)$ (in traditional relu $\lambda = 0$). The size of conv filters used here are all 5×5, the filter numbers from the first to seventh convolution layers are 32, 64, 128, 256, 256, 256, 256. The last layer is a fully connected hidden layer with softmax nonlinear. The probability of each class is produced as:

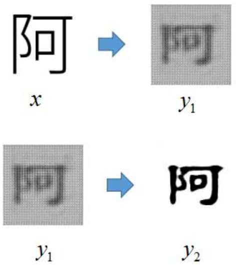

Figure 4. There are two steps in our generator:
1. $\boldsymbol{y_1} = g^1_{W_1}(\boldsymbol{x})$ we generate an image with target style and content but is not clear.
2. $\boldsymbol{y_2} = g^2_{W_2}(\boldsymbol{y_1})$ we generate an image with target style and content and is clear.

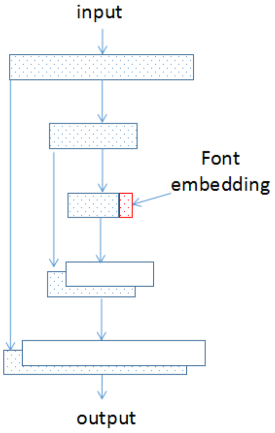

Figure 5. The sketch of our generator.

$$p_i = \frac{e^{z_i}}{\sum_{k=1}^{K} e^{z_k}} \tag{1}$$

Here, K is the number of classes which is 3755, z is the output of fully connected hidden layer, p_i is the probability of i-th class, we define the loss function as:

$$L_R = -\sum_{i=1}^{K} l_i log(p_i) \tag{2}$$

here, l is the labels of classes which is a one-hot vector.

3.3. Generator

The training of generator is aiming at font style transfer, the input is Chinese characters of Source Han Sans font style, the output is other fonts. In output layer, each pixel is not equally important, so if we define the loss directly on the output of the generator, the result will be averaged and some important parts are therefore missed.

Taking above problems into account, we employ the pre-training recognizer to help guide the learning procedure of generator. We first extract different feature maps from each layer of the pre-training recognizer. It has been proved that the lower feature maps represent more likely style features, while the higher feature maps represent more likely content features [5]. In this way, we can find a balance in the target style and target content. The structure of the generators is a common encoding-decoding structure [8], which is often used in image style transfer and image processing (*e.g.* [16], [17], [11]). As shown in Figure 5, our encoding-decoding is u-net ([19], [9]). Since the encoding network and the decoding network are symmetrical, and the size of their symmetric feature maps is the same, we can connect the pair of feature layers directly. The advantage by doing this is that, in the structure of encoding and decoding, we can not only pass the low-dimensional information, but also pass the information of image directly to the output. The encoding is also conv + bn + lrelu structure. After encoding procedure has been done, we concatenate the encoding output with the font embedding which is a $(N+1)$-dimensional vector. The left N dimensions are stored for train set while the remain 1 dimension is stored for test set. When testing, we can use this free dimension to finetune our generator. Decoding is also conv + bn + relu structure which is same as the encoding but with relu. Our input is 3D tensor 128 × 128 × 1, passing through 7 convolution operations (the numbers of convolution filters are 32, 64, 128, 256, 256, 256, 256), the 3D tensor is turned into 1 × 1 × 256. We concatenate this feature map with font embedding, then apply seven deconvolution operations (the number of the deconvolution filters are 256, 256, 256, 128, 64, 32, 1), the size of output is then turned into 128 × 128 × 1. We do not define loss directly on the output of generator [2] as:

$$arg \min_{W} [|g_W(\boldsymbol{x}) - \boldsymbol{t}|_1] \tag{3}$$

, because this will generate unreadable characters (shown in Figure 2). We take the output of generator as the input of the pre-training recognizer then define loss on the intermediate layers of R and the output layer of G rather than only the output layer of G. And the The generator is divided into two steps to generate the target font (Figure 4). As a result, in this way, each time we can get part features of the target fonts [23], the output image will be more clear. The sketch of the two generator is same, but we define different loss in each step.

3.4. G Learning with R

Step-1:We define generator-1 as $g^1_{W_1}$, W_1 is the parameters of generator-1, $\boldsymbol{x}$ is input of generator-1 which is Chinese characters in source font style, $\boldsymbol{y_1} = g^1_{W_1}(\boldsymbol{x})$ is output of generator-1, $\boldsymbol{t}$ is the target font style. We define the optimization function as:

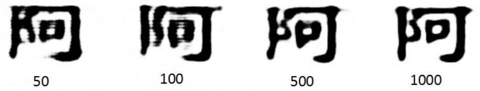

Figure 6. This figure show the result of using different number of characters in target font to finetune the generator.

$$arg \min_{W_1} \left[\sum_{i=1}^{N} \lambda_i \zeta_i (g^1_{W_1}(\boldsymbol{x}), \boldsymbol{t}) \right] \tag{4}$$

$\zeta_i(\boldsymbol{a}, \boldsymbol{b})$ is the regression loss defined on the i-th layer of recognizer. It is computed as Euclidean distance between feature representations:

$$\zeta_i(\boldsymbol{a}, \boldsymbol{b}) = \frac{1}{C_i H_i Q_i} \|r_i(\boldsymbol{a}) - r_i(\boldsymbol{b})\|_2^2 \tag{5}$$

$C_i \times H_i \times Q_i$ is the shape of i-th feature maps from recognizer, r_i is the value of the i-th feature maps of recognizer after activation layer. In our experiment we define $N = 6$, now we define loss on the first six layers of recognizer, this step we can generate characters with both target style and content but is not clear (Figure 4).

Step-2: Similarly, we define generator-2 as $g^2_{W_2}$, W_2 is the parameters in generator-2, $\boldsymbol{y_1}$ is input of generator-2 which is output of generator-1, $\boldsymbol{y_2} = g^2_{W_2}(\boldsymbol{y_1})$ is output of generator-2, $\boldsymbol{t}$ is the target font, we define the optimization function as:

$$arg \min_{W_2} \left[\lambda \zeta_{N_2}(g^2_{W_2}(\boldsymbol{y_1}), \boldsymbol{t}) + \beta |g^2_{W_2}(\boldsymbol{y_1}) - \boldsymbol{t}|_1 \right] \tag{6}$$

ζ is defined in step-1, $|g^2_{W_2}(\boldsymbol{y_1}) - \boldsymbol{t}|_1$ is MAE between target and generation, MAE on the output image can generate more clear characters compare with MSE [2]. In our experiment $N_2 = 6$, Now we define loss on output layer of generator and the 6th layers of recognizer to generate characters with content and clarity (Figure 4).

3.5. R Learning with G

After the training of generator, we can generate more characters in target font. We then feed these characters into the pre-training R, the accuracy of recognition in target font will be improved. We see this as a new supervised adaptive approach.

4. Experiments

We used 3755 classes (Ievel-1 set of GB2312-80) in 24 fonts for training and testing (Figure 7). We use all 3755 classes in 20 fonts to train both of generator and recognizer.

Table 1. THE STEPS IN OUR TRAINING AND ADAPTATION.

Recognizer	Generator
Train:	
Pretrain a recognizer.	
	Train generator with the help of pre-training R.
Adaptation:	
	Finetune generator with different number of subset of testing set in target font, then generate other characters in target font.
Finetune recognizer with the characters generated by generator in target font.	

Table 2. THE RESULT OF ADAPTATION IN FOUR FONTS.

#	Baseline	Adaptation-1	Adaptation-2
50	98.84%	98.84%	99.45%
100		98.87%	99.56%
500		98.91%	99.71%
1000		99.16%	99.74%
50	61.99%	78.40%	88.58%
100		81.05	90.57%
500		88.84%	94.81%
1000		90.76%	95.57%
50	79.16%	82.24%	87.10%
100		81.30%	88.91%
500		84.27%	89.78%
1000		86.30%	91.23%
50	27.53%	28.04%	49.23%
100		28.26%	51.70%
500		43.33%	62.86%
1000		49.92%	63.58%

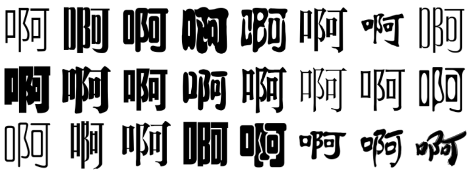

Figure 7. Example of Chinese character in 24 different fonts.

As for the remaining 4 fonts in test set, we randomly partition the 3755 classes into 1000 and 2755 classes, in which the 2755 classes are for testing. As for the other 1000 class, we will randomly select different number of characters in target font to finetune the generator and the recognizer. The detailed steps are shown in Table 1. All the models were implemented under the TensorFlow [1], platform using the NVIDIA Tesla K40m.

4.1. Experiments on Generator

Using different numbers of Chinese character samples in target font style to fintune the generator will get different result. The more the number of added samples, the generated image will be more clear. Figure 6 shows the results obtained by finetuning the generator with 50, 100, 500, and 1000 characters in target fonts.

4.2. Experiments on Recognizer

The characters in a specifical font style, which are generated by the generator are used to help finetuning the recognizer. This can be seen as a new supervised adaptation method. Different qualities of the generated characters will result in different improvement of the recognizer. If the generated characters are more like the character in the target style, the recognizer will therefore get better results. Since our adaptation method is supervised, for the sake of fairness, we tested our recognizer in three cases: recognizer don't see the target style at all (baseline), recognizer only see the characters used to finetune the generator (Adaptation-1), and recognizer see all characters in the target style generated by the generator (Adaptation-2) (See Table 2 we separately tested our recognizer with 50, 100, 500 and 1000 characters). When we use a smaller number of the characters in target fonts to adapt the generator and the recognizer, the improvement of our adaptive method is very noticeable compared to using only a smaller number of characters without the augmentation by generator in target fonts to finetune recognizer. if the number is 50, the performance of recognizer will be improved obviously separately increased by 0.61%, 10.18%, 4.85%, 21.19% compared Adaptation-1 with no augmentation by generator. The greater difference of font between the testing set and the training set, the more obvious the increase.

5. Conclusions

In this paper, we present a dual learning method to optimize our Chinese character recognizer and Chinese character generator. They help with each other during the overall training procedure. The recognizer helps the generator grab the key of target font, the generator helps the recognizer by predicting the character in the special style font. The results show that the system has a big improvement compare with traditional method which have no interaction between recognizer and generator.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China under Grants 61671422 and U1613211, in part by the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant XDB02070006, and in part by the National Key Research and Development Program of China under Grant 2017YFB1002200.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016. 5

[2] S. Baluja. Learning typographic style. *arXiv preprint arXiv:1603.04000*, 2016. 1, 2, 4

[3] D. Ciregan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3642–3649. IEEE, 2012. 2

[4] J. Du and Q. Huo. A discriminative linear regression approach to adaptation of multi-prototype based classifiers and its applications for chinese ocr. *Pattern Recognition*, 46(8):2313–2322, 2013. 2

[5] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2414–2423, 2016. 4

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 2

[7] B. Graham. Spatially-sparse convolutional neural networks. *arXiv preprint arXiv:1409.6070*, 2014. 3

[8] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006. 4

[9] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint arXiv:1611.07004*, 2016. 4

[10] T. Kim, M. Cha, H. Kim, J. Lee, and J. Kim. Learning to discover cross-domain relations with generative adversarial networks. *arXiv preprint arXiv:1703.05192*, 2017. 2

[11] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 4

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 2, 3

[13] C. Li and M. Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *European Conference on Computer Vision*, pages 702–716. Springer, 2016. 2

[14] Z. Lian, B. Zhao, and J. Xiao. Automatic generation of large-scale handwriting fonts via style learning. In *SIGGRAPH ASIA 2016 Technical Briefs*, page 12. ACM, 2016. 1

[15] T. Miyazaki, T. Tsuchiya, Y. Sugaya, S. Omachi, M. Iwamura, S. Uchida, and K. Kise. Automatic generation of typographic font from a small font subset. *arXiv preprint arXiv:1701.05703*, 2017. 1, 2

[16] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2536–2544, 2016. 4

[17] G. Perarnau, J. van de Weijer, B. Raducanu, and J. M. Álvarez. Invertible conditional gans for image editing. *arXiv preprint arXiv:1611.06355*, 2016. 4

[18] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 3

[19] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer, 2015. 4

[20] Y. Tian. zi2zi. *https://github.com/kaonashi-tyc/zi2zi*, 2017. 2

[21] C. Wu, W. Fan, Y. He, J. Sun, and S. Naoi. Handwritten character recognition by alternately trained relaxation convolutional neural network. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 291–296. IEEE, 2014. 2, 3

[22] S. Xu, T. Jin, H. Jiang, and F. C. Lau. Automatic generation of personal chinese handwriting by capturing the characteristics of personal handwriting. In *IAAI*, 2009. 1

[23] H. Zhang, T. Xu, H. Li, S. Zhang, X. Huang, X. Wang, and D. Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. *arXiv preprint arXiv:1612.03242*, 2016. 2, 4

[24] J. Zhang, J. Du, S. Zhang, D. Liu, Y. Hu, J. Hu, S. Wei, and L. Dai. Watch, attend and parse: An end-to-end neural network based approach to handwritten mathematical expression recognition. *Pattern Recognition*, 2017. 3

[25] X.-Y. Zhang, Y. Bengio, and C.-L. Liu. Online and offline handwritten chinese character recognition: A comprehensive study and new benchmark. *Pattern Recognition*, 61:348–360, 2017. 1, 2, 3

[26] X.-Y. Zhang, F. Yin, Y.-M. Zhang, C.-L. Liu, and Y. Bengio. Drawing and recognizing chinese characters with recurrent neural network. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 2

[27] Z. Zhong, L. Jin, and Z. Feng. Multi-font printed chinese character recognition using multi-pooling convolutional neural network. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 96–100. IEEE, 2015. 1, 2

[28] Z. Zhong, L. Jin, and Z. Xie. High performance offline handwritten chinese character recognition using googlenet and directional feature maps. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 846–850. IEEE, 2015. 2, 3