2019 International Conference on Document Analysis and Recognition (ICDAR)

# A Modified Inception-ResNet Network with Discriminant Weighting Loss for Handwritten Chinese Character Recognition

Linhui Chen, Liangrui Peng, Gang Yao, Changsong Liu and Xudong Zhang
*Beijing National Research Center for Information Science and Technology*
*Department of Electronic Engineering, Tsinghua University, Beijing, China*
*Email:chenlh16@gmail.com, {penglr, lcs}@tsinghua.edu.cn, {g-yao15, zhangxd18}@mails.tsinghua.edu.cn*

***Abstract*—Handwritten Chinese character recognition (HCCR) is a representative large character set pattern classification task. Recently, convolutional neural networks have provided promising solutions for this challenging task. This paper adopts the modified Inception-ResNet network for handwritten Chinese character recognition, and proposes a discriminant weighting method for cross-entropy loss calculation which focuses on recognition errors in the training stage. Sparse training technique is also incorporated. Under the specific condition of utilizing the testing mini-batch mean and variance for batch normalization, the proposed method achieves improved performance on the ICDAR-2013 offline handwritten Chinese character competition dataset.**

## I. Introduction

Handwritten Chinese character recognition (HCCR) has been widely applied in automatic recognition and processing of different types of documents. In the past decades, numerous works have been conducted on improving the accuracy of offline HCCR system. However, due to the large character set, free writing styles and shape similarities of Chinese characters, offline HCCR still remains a challenging task.

Earlier traditional machine learning (ML) based solutions for offline HCCR were often complicated for the multi-step structure (shape normalization [1] [2], feature extraction [3] and classification [4] [5]). With the rapid development of deep learning technology in recent years, convolutional neural network (CNN) based methods have dominated various image classification tasks. It has been proved that the CNN-based methods for HCCR present significant higher recognition performance than the traditional ones. The first CNN model for HCCR was proposed by Ciresan et al. [6]. In ICDAR 2013 offline HCCR competition [7], an alternately trained relaxation CNN model proposed by the team from Fujitsu [8] won the first place with a 94.77% accuracy.

Many CNN structures have been used for HCCR. Zhong et al. [9] proposed a GoogLeNet [10] based HCCR method, which outperformed the human performance for the first time. Deep residual networks [11] and their variants have achieved much higher accuracy than pervious CNN models [12] [13] [14] [15]. Szegedy et al. [16] proposed a novel Inception-ResNet structure, which used the inception modules to substitute the original convolutional blocks in ResNet. It has been proved that Inception-ResNet structure can accelerate the convergence speed and improve the model accuracy in many image classification tasks.

Loss functions are also important in CNN based methods. The center loss based metric learning for HCCR was utilized in [14]. By combining the triplet loss and softmax loss, a more accurate CNN model was proposed in [13].

More accurate hybrid CNN models adopted auxiliary feature information. Radical information has been used to improve the performance of HCCR [17] [18]. The normalization-cooperated direction-decomposed feature map and writer adaptation method were integrated with CNN model to achieve an effective system [19]. Traditional MQDF classifier was also utilized after CNN feature extractor to achieve a hybrid HCCR method [20]. The adversarial feature learning method [21] by exploring writer-independent semantic features under the guidance of printed characters has achieved an accuracy of 98.29% on the ICDAR 2013 offline HCCR competition dataset.

As for CNN model training stage, Han et al. [22] proposed a three-stage training strategy named Dense-Sparse-Dense (DSD) training for CNN. The core step was the sparse training. Unlike the simple dropout method [23] which aimed at deleting connections randomly, sparse training focused on pruning the low-weight connections and obtaining a sparse model for training.

The main contributions of our work are summarized as follows:

1) A modified Inception-ResNet model is utilized for offline HCCR.
2) A discriminant weighting method for cross-entropy loss calculation is proposed which focuses on recognition errors in the training stage.
3) Specific batch normalization and sparse training are also utilized to achieve a higher recognition accuracy.

The remaining parts of the paper are organized as follows: Section II gives the details of our proposed method. Experimental results are described and discussed in Section III, and the conclusion is presented in Section IV.

2379-2140/19/$31.00 ©2019 IEEE
DOI 10.1109/ICDAR.2019.00197

## II. METHODS

### A. Network overview

The original Inception-ResNet network [16] is mainly composed of five different types of blocks: stem, reduction module, Inception-ResNet-A, Inception-ResNet-B and Inception-ResNet-C. The three different Inception-ResNet blocks only differ in the Inception module used in residual connection. The network described in [16] contains 5 Inception-ResNet-A blocks, 10 Inception-ResNet-B blocks and 5 Inception-ResNet-C blocks. The Inception-ResNet for HCCR in this paper is modified from the original Inception-ResNet network. There are mainly three modifications:

1) Only one Inception-ResNet-A block, one Inception-ResNet-B block and one Inception-ResNet-C block are utilized for efficiency.
2) Parameters of convolutional layer have been modified to adapt the input character size of $96 \times 96$, which is smaller than the default $299 \times 299$.
3) When testing our model on the testing dataset, we use the current testing mini-batch mean and variance for batch normalization instead of the accumulated global statistics of the training dataset. The specific structure of our modified Inception-ResNet for offline HCCR is shown in Table 1.

Table 1: Modified Inception-ResNet network architecture.

| Layer Name | Type | Output Shape |
|---|---|---|
| Input | 96×96 grayscale image | 96×96×1 |
| Conv1 | 3×3 Conv stride 2, BN, ReLU | 48×48×32 |
| Conv2 | 3×3 Conv stride 1, BN, ReLU | 48×48×32 |
| Conv3 | 3×3 Conv stride 1, BN, ReLU | 48×48×64 |
| MaxPool1 | 3×3 max-pool stride 2 | 24×24×64 |
| Conv4_reduce | 1×1 Conv stride 1, BN, ReLU | 24×24×80 |
| Conv4 | 3×3 Conv stride 1, BN, ReLU | 24×24×192 |
| MaxPool2 | 3×3 max-pool stride 2 | 12×12×192 |
| Inception | 4-branch Inception | 12×12×320 |
| Inception-ResNet-A | 1×Inception-ResNet-A | 12×12×320 |
| Reduction-A | Inception module with downsampling | 6×6×1088 |
| Inception-ResNet-B | 1×Inception-ResNet-B | 6×6×1088 |
| Reduction-B | Inception module with downsampling | 3×3×2080 |
| Inception-ResNet-C | 1×Inception-ResNet-C | 3×3×2080 |
| Conv5 | 1×1 Conv stride 1, BN, ReLU | 3×3×1536 |
| AvePool | global max-pool | 1×1×1536 |
| Dropout | dropout-ratio 0.2 | 1×1×1536 |
| InnerProduct | 3755-way fully-connected | 1×3755 |
| Output | 3755-way Softmax | 3755 |

### B. Discriminant weighting loss

The softmax cross-entropy loss has been widely adopted in the training process of various deep neural networks. The original softmax cross-entropy loss $L_0$ can be formulated as:

$$p_j = \frac{e^{z_j}}{\sum_{k=0}^{K-1} e^{z_k}} \qquad j = 0, ..., K-1 \tag{1}$$

$$L_0 = -log(p_i) \tag{2}$$

$$\frac{\partial L_0}{\partial z_j} = \begin{cases} p_j, & j \neq i \\ p_j - 1, & j = i \end{cases} \tag{3}$$

where $\{z_j\}, j = 0, ..., K-1$ are the output values of the last fully-connected layer, $\{p_j\}, j = 0, ..., K-1$ are the predicted class probabilities output by the softmax layer and $i$ denotes the ground truth class label of the input sample.

Such a loss function neither differentiates between hard and easy examples, nor does it take confusing classes into consideration. Hence there have been some variants or extensions of the standard cross-entropy loss for specific application scenarios, such as the focal loss [24].

*1) Intuition:* In practice, there has always been a gap between the top-1 accuracy and the top-5 accuracy of a trained classification model. The confusing classes usually have higher scores of the predicted probabilities than the ground truth class. In order to focus on the confusing classes when there are recognition errors, we propose a novel loss function with a discriminant weighting factor.

*2) Definition:* We define the discriminant weighting loss as:

$$L = \alpha L_0 \tag{4}$$

where

$$\alpha = 1 + \lambda c \tag{5}$$

is the discriminant weighting factor. The confusion coefficient $c$ is obtained by counting the quantity of the confusing classes which have higher scores than the ground truth class. To control the influence of $c$, a hyper parameter $\lambda$ is adopted. Formally, we define the discriminant weighting loss as:

$$L = -(1 + \lambda c) log(p_i) \tag{6}$$

Its back propagation can be elaborated as Algorithm 1.

**Algorithm 1** Back propagation of the discriminant weighting loss.

**initialization**: $\frac{\partial L}{\partial z_j} = \frac{\partial L_0}{\partial z_j}, j = 0, ..., K-1$
$c \leftarrow 0$
**for** $j \leftarrow 0$ to $K-1$ **do**
  **if** $p_j \geq p_i$ **and** $j \neq i$ **then**
    $c \leftarrow c + 1$
  **end if**
**end for**
**for** $j \leftarrow 0$ to $K-1$ **do**
  $\frac{\partial L}{\partial z_j} \leftarrow (1 + \lambda c) \frac{\partial L}{\partial z_j}$
**end for**
**output** $\{\frac{\partial L}{\partial z_j}\}_{j=0}^{K-1}$

Different from the focal loss [24], which has the form of

$$L = -(1 - p_i)^{\gamma} log(p_i) \tag{7}$$

we do not adopt the model's estimated probability for the ground truth class in our discriminant weighting factor. As a minor change in the estimated probability may result in large difference in the focal loss value.

Intuitively, this discriminant weighting factor introduces the loss contribution from the confusing classes. We note two properties for our discriminant weighting loss.

1) When the confusion coefficient $c$ or the hyper parameter $\lambda$ is 0, the discriminant weighting factor is 1 and the loss is the original softmax cross entropy loss. As $c$ and $\lambda$ increase, the total loss increases. We can adjust the value of $\lambda$ to control the effect of the discriminant weighting factor.
2) Our discriminant weighting loss function gets larger loss values for recognition errors, which helps to improve the model generalization ability.

### C. Sparse training

Sparse training mechanism is introduced by implementing weight pruning during training. Weight pruning has been a practical method for reducing redundancy in large-scale parameters of deep neural networks. There can be up to hundreds of millions of parameters in a single model. Such a quantity of parameters often make it hard to train the model. Besides, a high proportion of the parameters are of very small values, which contribute little to both forward and backward propagations. And these redundant parameters in turn interfere with the training process. For this reason, an appropriate sparse rate can always promote the classification accuracy growth.

To perform sparse training, we add a mask to distinguish between important and unimportant weights. During an iteration, the weights of every convolutional and fully connected layer are sorted by their absolute values. Afterwards the smallest parts are masked as unimportant connections, according to a specific sparse rate $r$. They are set to 0 and not updated in this iteration. The implementation can be interpreted by Equation (8) to (12).

$$s = sort(|W^i|) \tag{8}$$

$$\theta = s[r \cdot length(s)] \tag{9}$$

$$Mask = \mathbb{1}(|W^i| > \theta) \tag{10}$$

$$W^i = W^i \cdot Mask \tag{11}$$

$$W^{i+1} = W^i - \eta^i \nabla f(W^i; x^i) \cdot Mask \tag{12}$$

The superscript $i$ represents the $i$-th iteration. The parameter $\eta$ in Equation (12) stands for the learning rate while $x$ is the input. And the mask matrix takes effect by pointwise multiplication in both Equation (11) and (12), which is different from the DSD training method [22]. We have found that a sparse rate $r = 0.1$ can achieve better performance in our experiments.

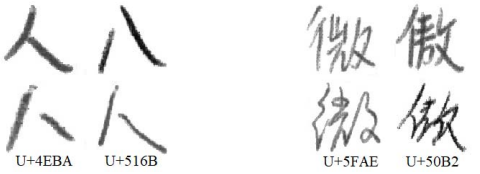
## III. Experiments

### A. Experiment settings

Our experiments are conducted without extra data preprocessing or any data augmentation skills. Our algorithms are implemented based on Caffe deep learning framework. We use NVIDIA TITAN Xp GPUs for parallel computation. The proposed loss and sparse training are performed after 5 epochs' training with the original loss. There are also experiments for comparisons of continuing training and training from scratch. Batch normalization and dropout are utilized in our network architecture. Since the handwritten Chinese character dataset is a large set with considerable class numbers, batch normalization helps to reshape the distributions of each layer's input while making the training process more efficient. And dropout improves our model's generalization ability. Both batch normalization and dropout contribute to avoid overfitting.

### B. Dataset

We use the CASIA offline handwritten Chinese character datasets HWDB 1.0-1.1 as the training dataset and the ICDAR-2013 offline competition dataset as the testing dataset. The handwritten character datasets include 3755 classes of Chinese characters. The training dataset is written by 576 volunteers and has over 2.4 million samples. The testing dataset is written by 60 volunteers and contains about 224 thousand samples. Some confusing samples from the training dataset are shown in Figure 1. The two images from the same column are of the same class. Obviously, not only the simple similar patterns but also some complex ones can be hard to differentiate due to irregular writing styles.




(a) simple patterns (b) complex patterns

Figure 1: Samples from the training dataset.

### C. Comparison of different $\lambda$ values for discriminant weighting loss

The parameter $\lambda$ in Equation (6) controls the weighting for the confusing classes. We explore the effects of different $\lambda$ values. The results for comparison are shown in Table 2. Our baseline method which uses the standard cross-entropy loss is the case where $\lambda$ is set to 0. With $\lambda$ as 0.01, we get the highest accuracy so far. This is due to the balance between the ground truth class and the confusing classes. And we

select this value in the following experiments with sparse training. There is not as much performance improvement in the top-5 accuracy as in the top-1 accuracy. This indicates that the proposed loss has successfully narrowed the gap between the top-1 and top-5 accuracy in our experiments.

Table 2: Recognition accuracy with different $\lambda$ values for continuing training when testing in the batch size of 100.

| $\lambda$ | Top-1 accuracy(%) | Top-5 accuracy(%) |
|---|---|---|
| 0 | 98.29 | 99.67 |
| 0.01 | **98.68** | **99.76** |
| 0.1 | 98.43 | 99.74 |

Table 3: Recognition accuracy with different $\lambda$ values for training from scratch when testing in the batch size of 100.

| $\lambda$ | Top-1 accuracy(%) | Top-5 accuracy(%) |
|---|---|---|
| 0 | 98.20 | 99.65 |
| 0.01 | **98.28** | **99.69** |
| 0.1 | 98.15 | 99.69 |

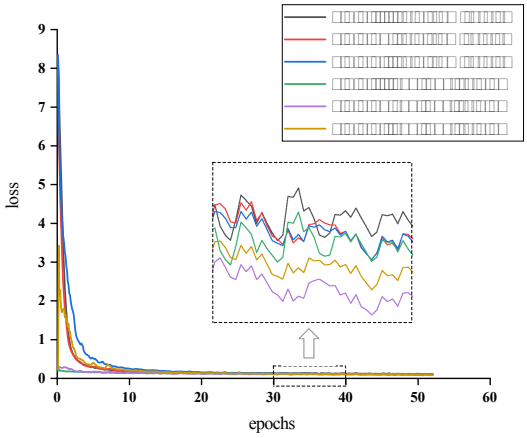



Figure 2: Training loss of both training strategies with different $\lambda$ values.

### *D. Comparison of continuing training and training from scratch*

When the discriminant weighting loss is used to train a character classification model from scratch, its performance may be slightly worse than the standard softmax cross-entropy loss. The results are in Table 3. As shown in Figure 2, continuing training with the proposed loss after several epochs' training with standard loss always surpasses training from scratch. This is probably because the discriminant weighting loss is designed for small discriminant weighting

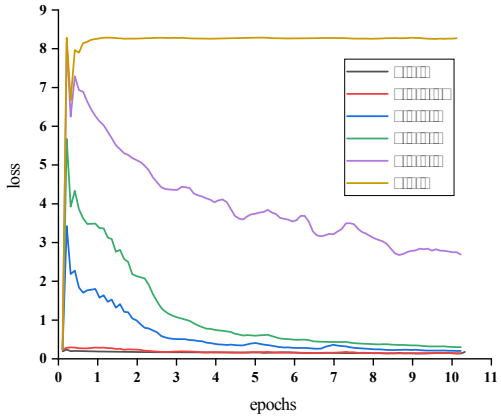



Figure 3: Training loss of continuing training with different $\lambda$ values.

factor values. In the beginning of the training process, the confusion coefficient in the discriminant weighting factor tends to be large, which may interfere with the parameter optimization process. This is much more in evidence on the occasions when $\lambda$ is too large. Figure 3 shows as $\lambda$ exceeds 0.1 and keeps on increasing, the training process is getting more and more unstable and the loss values are eventually unable to decline.

### *E. Sparse training*

Table 4 shows the results of utilizing sparse training with/without the proposed discriminant weighting loss (DWL). Our baseline method which uses the standard cross-entropy loss is the same as the case where $\lambda$ is set to 0 in Table 2. Performing pruning upon the baseline method leads to slight improvement in accuracy, with a sparse rate of 0.1 in our experiments. However, with the help of the proposed loss, the progress is more obvious. DWL combined with sparse training method achieves the accuracy of 98.79% when testing in the batch size of 100.

Table 4: Recognition accuracy with different training methods when testing in the batch size of 100.

| Method | Top-1 accuracy(%) | Top-5 accuracy(%) |
|---|---|---|
| Baseline | 98.29 | 99.67 |
| Baseline+Pruning | 98.32 | 99.68 |
| Baseline+Pruning+DWL | **98.79** | **99.80** |

### *F. Selection of testing batch size*

Since our model uses the testing mini-batch mean and variance for batch normalization, we also investigate the impact of different testing batch size. When batch size is set to 1, the testing mini-batch mean and variance do not make

Table 5: Comparison with other methods on ICDAR-2013 handwriting competition dataset.

| Model | Ref. | Recognition Accuracy(%) |
|---|---|---|
| Human Performance | [7] | 96.13 |
| Ensemble-DCNN-Similarity Ranking | [13] | 97.64 |
| HCCR-CNN12Layer | [25] | 97.59 |
| Melnyk-Net(Model C) | [26] | 97.61 |
| Adversarial Feature Learning | [21] | 98.29 |
| Inception-ResNet + DWL + Pruning | Ours | **98.79*** |

* Please note that our model achieves this accuracy under the specific condition of utilizing the testing mini-batch mean and variance for batch normalization. The testing batch size is 100.

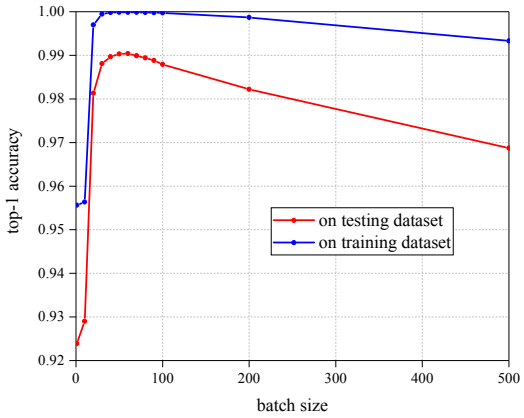



Figure 4: Recognition accuracy with different testing batch size.

sense, and the accumulated global statistics of the training dataset are used instead. Figure 4 shows the testing results on both the training set and the test set. As the batch size of 64 is used during training, the testing results on both the training set and the test set achieve the peaks when testing batch size is around 60. This indicates that the batch size values used in training and testing should match. To verify the effectiveness of using the testing mini-batch mean and variance for batch normalization, we have conducted additional experiments by dividing the test set into ten subsets. Each time we select one subset for batch size selection by testing the model on the selected subset with different batch size values, the selected batch sizes are still around 60 for these ten experiments.

## IV. Conclusion

In this paper, we propose a novel convolutional neural network based method for offline handwritten Chinese character recognition. The modified Inception-ResNet architecture is adopted. We propose the discriminant weighting loss to reduce the confusing classes and improve the accuracy. The sparse training method is combined to make further progress by weight pruning. Under the condition of testing in some specific batch size values, we achieve improved performance on the ICDAR-2013 offline handwritten character competition dataset. In the future, we will investigate our method's performance with more network structures and different ways of making the testing batch for batch normalization.

## Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper. This research is supported by National Key R&D Program of China (No. 2019QY1804) and National Natural Science Foundation of China (No. U1636124 and No. 61573028).

## References

[1] Truyen Van Phan, JinFeng Gao, Bilan Zhu, and Masaki Nakagawa. Effects of line densities on nonlinear normalization for online handwritten Japanese character recognition. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 834–838, 2011.

[2] Cheng-Lin Liu and Katsumi Marukawa. Pseudo two-dimensional shape normalization methods for handwritten Chinese character recognition. *Pattern Recognition*, 38(12):2242–2255, 2005.

[3] Cheng-Lin Liu. Normalization-cooperated gradient feature extraction for handwritten character recognition. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 29(8):1465–1469, 2007.

[4] Fumitaka Kimura, Kenji Takashina, Shinji Tsuruoka, and Yasuji Miyake. Modified quadratic discriminant functions and the application to Chinese character recognition. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 9(1):149–153, 1987.

[5] Cheng-Lin Liu, Hiroshi Sako, and Hiromichi Fujisawa. Discriminative learning quadratic discriminant function for handwriting recognition. *IEEE Transactions on Neural Networks*, 15(2):430–444, 2004.

[6] Dan Cireşan and Ueli Meier. Multi-column deep neural networks for offline handwritten Chinese character classification. In *Proceedings of the International Joint Conference on Neural Networks*, pages 1–6, 2015.

[7] Fei Yin, Qiu-Feng Wang, Xu-Yao Zhang, and Cheng-Lin Liu. ICDAR 2013 Chinese handwriting recognition competition. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 1464–1470, 2013.

[8] Chunpeng Wu, Wei Fan, Yuan He, Jun Sun, and Satoshi Naoi. Handwritten character recognition by alternately trained relaxation convolutional neural network. In *Proceedings of the International Conference on Frontiers in Handwriting Recognition*, pages 291–296, 2014.

[9] Zhuoyao Zhong, Lianwen Jin, and Zecheng Xie. High performance offline handwritten Chinese character recognition using GoogLeNet and directional feature maps. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 846–850, 2015.

[10] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[12] Zhao Zhong, Xu-Yao Zhang, Fei Yin, and Cheng-Lin Liu. Handwritten Chinese character recognition with spatial transformer and deep residual networks. In *Proceedings of the International Conference on Pattern Recognition*, pages 3440–3445, 2016.

[13] Cheng Cheng, Xu-Yao Zhang, Xiao-Hu Shao, and Xiang-Dong Zhou. Handwritten Chinese character recognition by joint classification and similarity ranking. In *Proceedings of the International Conference on Frontiers in Handwriting Recognition*, pages 507–511, 2016.

[14] Ruyu Zhang, Qingqing Wang, and Yue Lu. Combination of ResNet and center loss based metric learning for handwritten Chinese character recognition. In *Proceedings of the International Conference on Document Analysis and Recognition*, volume 5, pages 25–29, 2017.

[15] Wenchao Wang, Jianshu Zhang, Jun Du, Zi-Rui Wang, and Yixing Zhu. DenseRAN for offline handwritten Chinese character recognition. In *Proceedings of the International Conference on Frontiers in Handwriting Recognition*, pages 104–109, 2018.

[16] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, Inception-ResNet and the impact of residual connections on learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 4278–4284, 2017.

[17] Yaming Sun, Lei Lin, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. Radical-enhanced Chinese character embedding. In *Proceedings of the International Conference on Neural Information Processing*, pages 279–286, 2014.

[18] Luo Weike and Kamata Sei-Ichiro. Radical region based CNN for offline handwritten Chinese character recognition. In *Proceedings of the Asian Conference on Pattern Recognition*, pages 542–547, 2017.

[19] Xu-Yao Zhang, Yoshua Bengio, and Cheng-Lin Liu. Online and offline handwritten Chinese character recognition: A comprehensive study and new benchmark. *Pattern Recognition*, 61:348–360, 2017.

[20] Yanwei Wang, Xin Li, Changsong Liu, Xiaoqing Ding, and Youxin Chen. An MQDF-CNN hybrid model for offline handwritten Chinese character recognition. In *Proceedings of the International Conference on Frontiers in Handwriting Recognition*, pages 246–249, 2014.

[21] Yaping Zhang, Shan Liang, Shuai Nie, Wenju Liu, and Shouye Peng. Robust offline handwritten character recognition through exploring writer-independent features under the guidance of printed data. *Pattern Recognition Letters*, 106:20–26, 2018.

[22] Song Han, Jeff Pool, Sharan Narang, Huizi Mao, Enhao Gong, Shijian Tang, Erich Elsen, Peter Vajda, Manohar Paluri, John Tran, et al. DSD: Dense-sparse-dense training for deep neural networks. *arXiv preprint arXiv:1607.04381*, 2016.

[23] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[24] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. *arXiv preprint arXiv:1708.02002*, 2017.

[25] Xuefeng Xiao, Lianwen Jin, Yafeng Yang, Weixin Yang, Jun Sun, and Tianhai Chang. Building fast and compact convolutional neural networks for offline handwritten Chinese character recognition. *Pattern Recognition*, 72:72–81, 2017.

[26] Pavlo Melnyk, Zhiqiang You, and Keqin Li. A high-performance CNN method for offline handwritten Chinese character recognition and visualization. *arXiv preprint arXiv:1812.11489*, 2018.