Supplementary Material for
Zero-Shot Chinese Character Recognition with Stroke-Level Decomposition

Paper ID 819

Contents

A Detailed configuration

A.1 Configuration of the image-to-feature encoder

The configuration of the image-to-feature encoder is demonstrated in Table 1.

Layer Name	Configuration
CNN	*in*=3, *out*=64
Max-Pooling	*kernel*=2, *stride*=2
CNN	*in*=64, *out*=128
Building Block	*number*=3, *in*=128, *out*=256
Building Block	*number*=4, *in*=256, *out*=256
Building Block	*number*=6, *in*=256, *out*=512
Building Block	*number*=3, *in*=512, *out*=1024

Table 1: Details of the image-to-feature encoder. *in*, *out*, *kernel*, *stride*, *number* denote input channel, output channel, kernel size, stride size, and the number of blocks, respectively. The sizes of kernel, stride, padding in CNN are set to 3, 1, 1 as default. A batch normalization layer and a ReLU layer is added after each CNN.

Since the input image is quite small (32×32), we only utilize one max-pooling layer in the encoder. To study the impact on the size of square feature maps, we sequentially add a max-pooling layer before the first three building blocks to adjust the output size ranging from $\{8, 4, 2\}$. Moreover, we also conduct an experiment by removing the only max-pooling resulting in an output size of 32, which is equal to the original input. The experimental setting follows: From HWDB1.0-1.1, we choose samples with labels in the first 1500 classes as the training set. From ICDAR2013, we choose samples with labels in the last 1000 classes as the test set. The results are shown in Table 2. The performance reaches the best with the feature size 16. The reasons can be concluded in two perspectives: 1) The size of feature maps will be further reduced as more max-pooling layers are appended, thus resulting in the loss of key information. 2) If we remove all the max-pooling layers, the redundant features will hamper our model from converging better.

Size	32	16	8	4	2
CACC	17.50%	**22.88%**	11.87%	4.87%	3.02%

Table 2: Ablation study on the size of square feature maps. For example, "32" means no max-pooling layers are used in the encoder and "2" means there is a max-pooling layer before the first three building blocks.

A.2 Configuration of the feature-to-stroke decoder

The configuration of the feature-to-stroke decoder is demonstrated in Figure 1. It mainly consists of three components, including the masked multi-head attention module (Masked MHA), the multi-head attention module (MHA), and the feed-forward module. Since Transformer has been widely used in both natural language processing (NLP) and computer vision (CV) tasks, we omit the exhaustive background description of the model architecture and refer readers to the original Transformer [Vaswani *et al.*, 2017]. We empirically set the hyperparameters of the feature-to-stroke decoder following HolisticOCR [Yang *et al.*, 2020], which is demonstrated in Table 3.

Hypermeter	Value
The number of decoder blocks	1
Head number	4
Dimension of positional encoding	512
Dimension of embedding	512

Table 3: Hyperparameters and the corresponding values in the feature-to-stroke decoder.

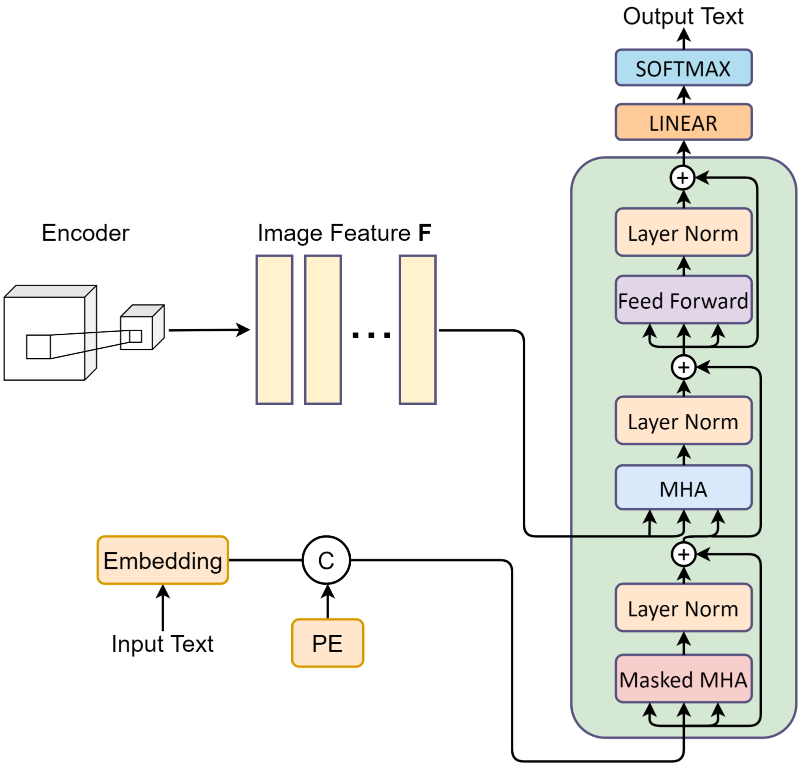

Figure 1: The architecture of the feature-to-stroke decoder. It follows the basic design of the original Transformer decoder.

B Division strategy of datasets

B.1 Printed artistic characters

3,755 Level-1 common-used characters are used as candidates during testing.

Character zero-shot experiment. We denote the whole set of printed artistic characters (394,275 samples) as $\mathcal{S}_{\text{PAC}}$. From $\mathcal{S}_{\text{PAC}}$, we choose samples with labels in the first m classes of 3,755 characters as the training set, where m ranges in $\{500,1000,1500,2000,2755\}$, and choose samples with labels in the last 1000 classes as the test set. 3,755 Level-1 characters are chosen as candidates during testing.

Radical zero-shot experiment. The setup follows three steps: 1) Calculate the frequency of each radical in 3,755 Level-1 characters. 2) If a character contains a radical that appears less than n times ($n \in \{50,40,30,20,10\}$), move it into set $\mathcal{S}_{\text{TEST}}$; otherwise, move it into $\mathcal{S}_{\text{TRAIN}}$. 3) Select all samples with labels in $\mathcal{S}_{\text{TRAIN}}$ from $\mathcal{S}_{\text{PAC}}$ as the training set, and all samples with labels in $\mathcal{S}_{\text{TEST}}$ from $\mathcal{S}_{\text{PAC}}$ as the test set.

B.2 Scene characters CTW

3,650 characters in CTW are used as candidates during testing.

Character zero-shot experiment. We denote the whole set of CTW (812,872 samples) as $\mathcal{S}_{\text{CTW}}$. Firstly, we sort the 3,650 classes that appeared in the CTW dataset [Yuan *et al.*, 2019] from front to back according to their positions in the national standard GB18030-2005. Then we collect the samples in the first 500 classes from $\mathcal{S}_{\text{CTW}}$ as the test set, and the following m classes from $\mathcal{S}_{\text{CTW}}$ as the training set, where $m \in \{500, 1000, 1500, 2000, 3150\}$. 3,650 characters that appear in CTW are chosen as candidates during testing.

Radical zero-shot experiment. The setup follows three steps: 1) Calculate the frequency of each radical in 3,755 Level-1 characters. 2) If a character contains a radical that appears less than n times ($n \in \{50,40,30,20,10\}$), move it into set $\mathcal{S}_{\text{TEST}}$; otherwise, move it into $\mathcal{S}_{\text{TRAIN}}$. 3) Select all samples with labels in $\mathcal{S}_{\text{TRAIN}}$ from $\mathcal{S}_{\text{CTW}}$ as the training set, and all samples with labels in $\mathcal{S}_{\text{TEST}}$ from $\mathcal{S}_{\text{CTW}}$ as the test set.

Recognition on full set of CTW. We employ the official training set of CTW for training (760,107 samples) and the official test set for validating (52,765 samples), which follows the same way as HDE [Cao *et al.*, 2020].

C Additional experimental setting and results

C.1 Relationship between *n* and training class in radical zero-shot experiments

As shown in Table 4, with the decrease of n, the capacity of the training set increases, which results in a higher accuracy.

n	Train class	Test class
50	839	2,916
40	1,170	2,585
30	1,646	2,109
20	2,146	1,609
10	3,019	736

Table 4: Relationship between *n* and training class in radical zero-shot experiments.

C.2 Experimental results of using the same encoder and decoder as DenseRAN

For a fair comparison, we employ the same encoder (DenseNet) and decoder (RNN) that used in DenseRAN [Wang *et al.*, 2018]. As shown in Table 5, our proposed method still outperforms the compared methods using different combinations of encoder and decoder.

Handwritten	m for Character Zero-Shot				
	500	1000	1500	2000	2755
DenseRAN [2018]	1.70%	8.44%	14.71%	19.51%	30.68%
HDE [2020]	4.90%	12.77%	19.25%	25.13%	33.49%
Ours (DenseNet + RNN)	4.23%	12.84%	21.85%	26.99%	35.38%
Ours (DenseNet + Transformer)	3.16%	12.66%	21.88%	**28.48%**	35.71%
Ours (ResNet + Transformer)	**5.60%**	**13.85%**	**22.88%**	25.73%	**37.91%**

Table 5: Character zero-shot experiment on handwritten characters.

C.3 Experimental results on characters not in confusable set

We pick out 207,675 characters that are not in in the confusable set $\mathcal{C}$ from ICDAR2013 for validating. The experimental results are shown in Table 6. The three methods utilize the same ResNet as the encoder and Transformer as the decoder for a fair comparison.

Method	Character-based	Radical-based	Stroke-based (Ours)
CACC	96.70%	96.78%	**96.93%**

Table 6: The experiment on those samples not in the confusable set.

D Results analysis

We first study the distribution of stroke-sequence lengths for the 3,755 Level-1 common-used characters (see Figure 2). The lengths range from $[1, 24]$ and over half of sequences has lengths in $[7, 11]$.

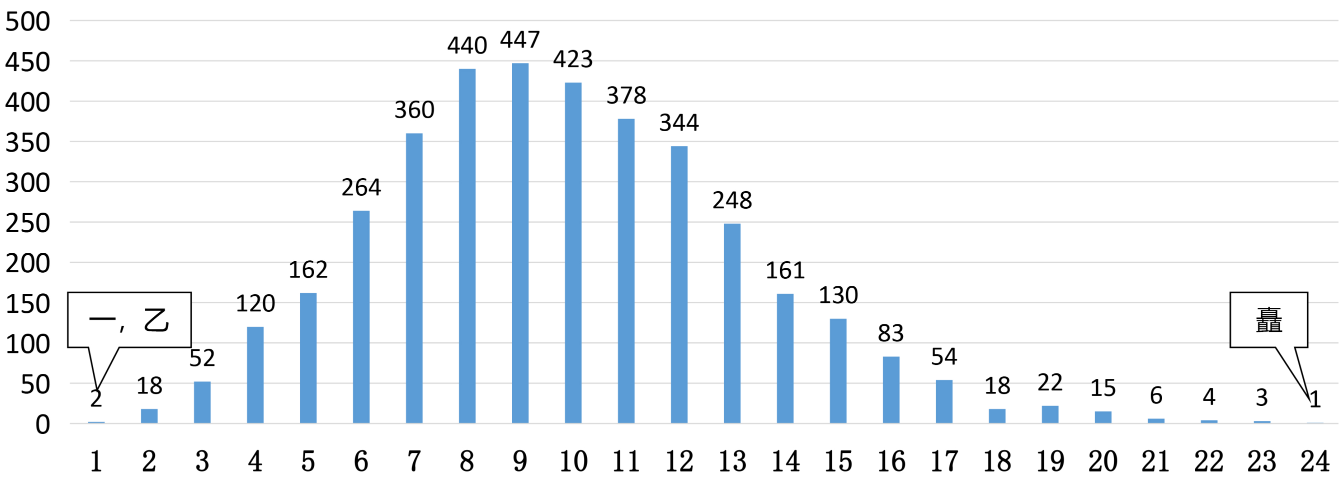

Figure 2: The distribution of stroke-sequence lengths for the 3,755 level-1 common-used characters.

We investigate the accuracy of different stroke-sequence lengths in terms of whether to use the lexicon rectification. We use the full set of HWDB1.0-1.1 [Liu *et al.*, 2013] dataset for training and the ICDAR2013 [Yin *et al.*, 2013] dataset for testing. Results are demonstrated in Figure 3. Interestingly, we observe that the longer the stroke sequences, the higher the accuracy. It may derive from two reasons: 1) A longer stroke sequence provides sufficiently context for the feature-to-stroke decoder. Therefore, minor errors can be corrected. 2) There are fewer samples with long stroke sequences, so the lexicon rectification module suffers less disturbances.

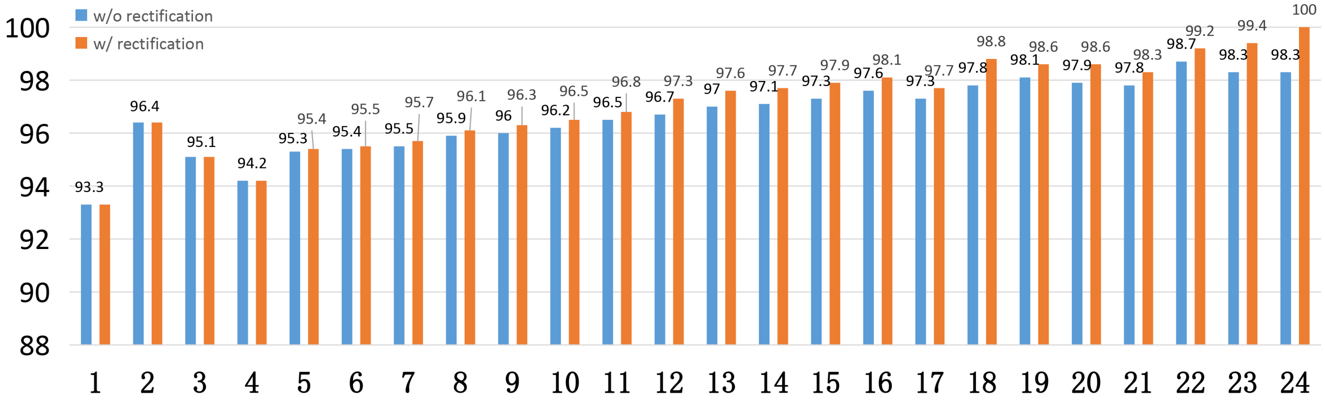

Figure 3: Accuracy for samples with different stroke sequence lengths and the effect of lexicon rectification.

E Discussion on class imbalance problem

To study the class imbalance problem, we investigate two datasets including HWDB1.0-1.1 [Liu *et al.*, 2013] and CTW [Yuan *et al.*, 2019]. Since the frequency of each character in HWDB1.0-1.1 is manually controlled at around 700, we only discuss the frequency of radicals and strokes in this dataset (see Figure 4). For the CTW dataset, since it derives from natural scenes, it can better reflect the class imbalance problem (see Figure 5). As is demonstrated in Table 7, we employ *imbalance ratio*, which is calculated by the ratio of the maximum frequency and the minimum frequency, to measure the degree of imbalance. Its value ranges from $[1, +\infty)$. A smaller imbalance ratio indicates a more balanced data distribution. We observe that the radical-level decomposition leads to a more severe class imbalance problem. On the contrary, our proposed stroke-based method is benefited from two aspects: 1) The stroke-level decomposition allows a more balanced class distribution. 2) The frequency of each class is far more than that in class-level and radical-level, which helps the model converge better.

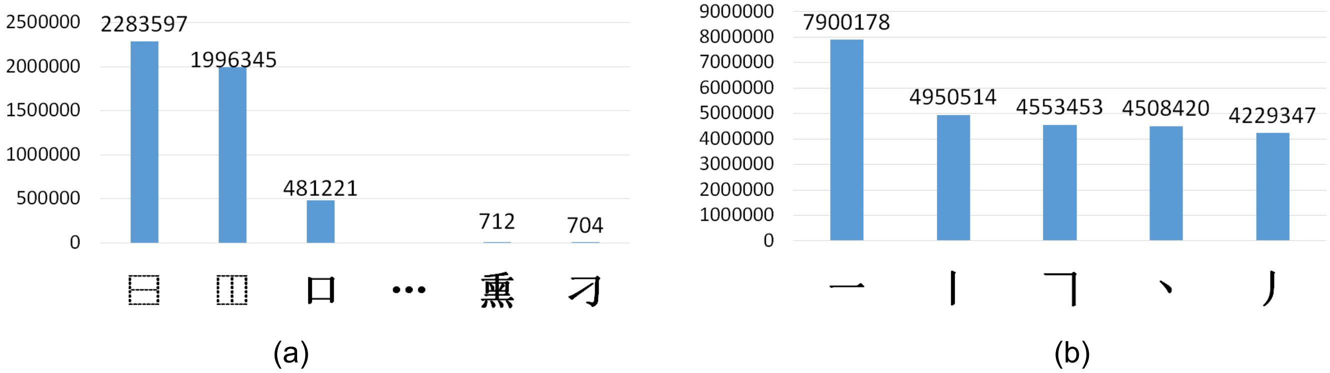

Figure 4: Class imbalance problem in the HWDB1.0-1.1 dataset. (a) Radical-level. (b) Stroke-level.

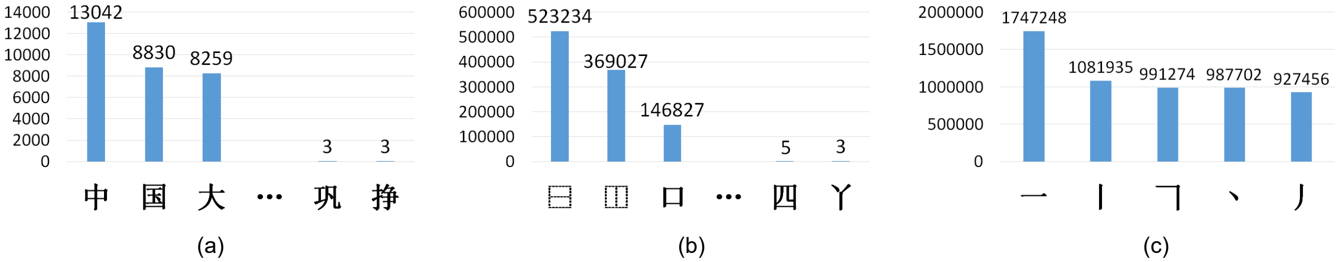

Figure 5: Class imbalance problem in the CTW dataset. (a) Character-level. (b) Radical-level. (c) Stroke-level.

Dataset	Level of Decomposition	Imbalance Ratio
HWDB1.0-1.1 [Liu *et al.*, 2013]	Radical	3,243.74
	Stroke	**1.87**
CTW [Yuan *et al.*, 2019]	Character	4,347.33
	Radical	174,411.33
	Stroke	**1.77**

Table 7: Imbalance ratio for each level in different datasets.

References

[Cao *et al.*, 2020] Zhong Cao, Jiang Lu, Sen Cui, and Changshui Zhang. Zero-shot handwritten chinese character recognition with hierarchical decomposition embedding. *PR*, 107:107–488, 2020.

[Liu *et al.*, 2013] Cheng-Lin Liu, Fei Yin, Da-Han Wang, and Qiu-Feng Wang. Online and offline handwritten chinese character recognition: benchmarking on new databases. *PR*, 46(1):155–162, 2013.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017.

[Wang *et al.*, 2018] Wenchao Wang, Jianshu Zhang, Jun Du, Zi-Rui Wang, and Yixing Zhu. Denseran for offline handwritten chinese character recognition. In *ICFHR*, pages 104–109, 2018.

[Yang *et al.*, 2020] Lu Yang, Peng Wang, Hui Li, Zhen Li, and Yanning Zhang. A holistic representation guided attention network for scene text recognition. *NC*, 414:67–75, 2020.

[Yin *et al.*, 2013] Fei Yin, Qiu-Feng Wang, Xu-Yao Zhang, and Cheng-Lin Liu. Icdar 2013 chinese handwriting recognition competition. In *ICDAR*, pages 1464–1470, 2013.

[Yuan *et al.*, 2019] Tai-Ling Yuan, Zhe Zhu, Kun Xu, Cheng-Jun Li, Tai-Jiang Mu, and Shi-Min Hu. A large chinese text dataset in the wild. *JCST*, 34(3):509–521, 2019.