# Decoupled Learning for Long-Tailed Oracle Character Recognition

Jing Li[1], Bin Dong[2], Qiu-Feng Wang[1(✉)], Lei Ding[2], Rui Zhang[3], and Kaizhu Huang[4]

[1] School of Advanced Technology, Xi'an Jiaotong-Liverpool University, Suzhou, China
Jing.Li19@student.xjtlu.edu.cn, Qiufeng.Wang@xjtlu.edu.cn
[2] Ricoh Software Research Center(Beijing) Co., Ltd., Beijing, China
{Bin.Dong,Lei.Ding}@cn.ricoh.com
[3] School of Mathematics and Physics, Xi'an Jiaotong-Liverpool University, Suzhou, China
Rui.Zhang02@xjtlu.edu.cn
[4] Data Science Research Center, Duke Kunshan University, Suzhou, China
Kaizhu.Huang@dukekunshan.edu.cn

**Abstract.** Oracle character recognition has recently made significant progress with the success of deep neural networks (DNNs), but it is far from being solved. Most works do not consider the long-tailed distribution issue in oracle character recognition, resulting in a biased DNN towards head classes. To overcome this issue, we propose a two-stage decoupled learning method to train an unbiased DNN model for long-tailed oracle character recognition. In the first stage, we optimize the DNN under instance-balanced sampling, obtaining a robust backbone but biased classifier. In the second stage, we propose two strategies to refine the classifier under class-balanced sampling. Specifically, we add a learnable weight scaling module which can adjust the classifier to respect tail classes; meanwhile, we integrate the KL-divergence loss to maintain attention to head classes through knowledge distillation from the first stage. Coupling these two designs enables us to train an unbiased DNN model in oracle character recognition. Our proposed method achieves new state-of-the-art performance on three benchmark datasets, including OBC306, Oracle-AYNU and Oracle-20K.

**Keywords:** Oracle character recognition · Long tail · Decoupled learning · Knowledge distillation

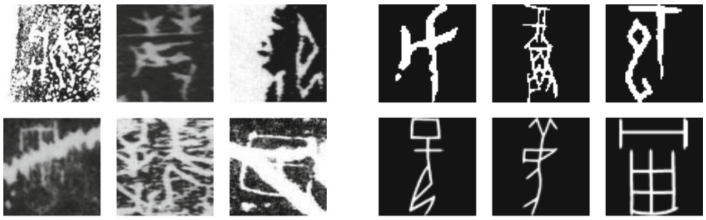
## 1 Introduction

As the earliest known writing system in China, the oracle bone script plays a significant role in archaeology, palaeography, and history. These characters were carved on turtle nails and animal bones for divination during the Shang Dynasty. Recognizing them usually requires a high level of expertise, which is both time-consuming and costly. Recently, much attention has been paid to investigating automatic recognition technologies for oracle characters. Such research has


G. A. Fink et al. (Eds.): ICDAR 2023, LNCS 14190, pp. 165–181, 2023.
https://doi.org/10.1007/978-3-031-41685-9_11

made great progress [1–3], but its performance still needs improvement so as to meet the requirements of practical applications. With the burst of deep neural networks (DNNs) and their successful applications in computer vision, natural language processing, etc [4], researchers have recently explored DNNs in oracle character recognition [2,3,5]. The success of DNN models usually needs a large size of labelled training samples. However, obtaining sufficient oracle character data is challenging due to the scarcity of sources and difficult labelling.

Thanks to the great efforts from the research community, there are some available oracle datasets, which can be divided into two categories: real rubbing character images and hand-copied character images. Some examples are shown in Fig. 1. As we can see, real images scanned from turtle nails and animal bones contain various noises, e.g., partially missing, dense white regions, and bone fractures. One public representative dataset is OBC306, collected by [2], which contains 309,511 character-level instances belonging to 306 classes. In contrast, hand-copied images are high-resolution images without noise, but it needs to invite experts to transcribe them. Two additional available datasets are Oracle-20K [1] and Oracle-AYNU [3]. Oracle-20K contains 19,491 character images and 249 classes, while Oracle-AYNU has 2,584 classes with 39,072 instances.



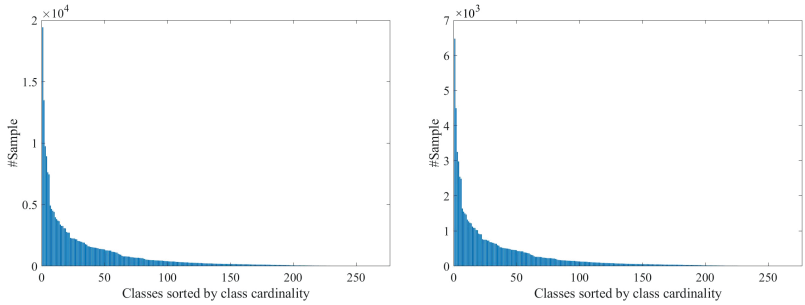
(a) Real rubbing images (b) Hand-copied images

**Fig. 1.** Examples of oracle character images.

Unfortunately, all these current oracle datasets suffer from the common issue arising in the long-tailed distribution, as shown in Fig. 2 for the OBC306 dataset. It is apparent that the number of samples varies significantly among classes. Specifically, in the training set of OBC306, the top five majority classes have over 10,000 instances while many classes have just one or two instances; in the test set of OBC306, around 17% of classes contain fewer than ten samples, while the largest class contains 6,474 samples. Such distribution leads the training of DNNs to suffer from a strong bias towards head classes; consequently, the learned model cannot learn robust classification for tail classes[1]. To address this issue, Zhang et al. [3] proposed a nearest neighbour classifier with metric learning for imbalanced oracle character recognition, and successfully improved the accuracy

[1] We divide the oracle data into three categories: the classes with many samples as head classes, the classes with few samples as tail classes, and the remainder are the medium classes described in Sect. 4.3.

on Oracle-AYNU and Oracle-20K. Furthermore, Li et al. [5] designed a mix-up strategy by combining softmax loss and triplet loss, and demonstrated the state-of-the-art performance on oracle datasets, including Oracle-20K, Oracle-AYNU, and OBC306. Albeit these advances, the issue of long-tailed distribution is still far from being solved in oracle character recognition.




(a) Training data distribution (b) Test data distribution

**Fig. 2.** Data distributions of OBC306.

Although the long-tailed issue has not yet received extensive attention in oracle character recognition, it has been intensively studied in general visual recognition tasks. For example, re-sampling training samples [6,7] or adjusting the loss value for each class [8,9] has been widely used to balance the class distribution during training. In addition, some methods [10,11] utilize label frequencies to shift output logits of models during training or post-processing. Some recent efforts have also been made to re-balance DNN models. In particular, studies [12–15] have shown that it is effective to decouple the one-stage training process into representation learning and classifier learning for imbalanced data. In general, such works adjust the classifier to focus on tail classes while sacrificing the performance of head classes during the training. They then validate the effectiveness on the test data with a balanced distribution across all classes. However, the test data in the oracle datasets is also heavily long-tailed as shown in Fig. 2. Performance sacrifice on head classes may unfortunately lead to the degradation of the total accuracy, though the average accuracy is still improved as presented in Table 2 of Sect. 4.4.

Motivated from the aforementioned analysis, we propose a two-stage-based decoupled learning method for long-tailed oracle character recognition, where the DNN model is split into a ViT [16] as the backbone network and a single fully connected layer network as the classifier. In the first stage, we train the DNN model with the standard cross-entropy loss under instance-balanced sampling. As the oracle data is limited, we utilize mixup augmentation [17] to exploit current oracle samples fully. Although a robust backbone model is learned in the first stage, the long-tailed oracle data distribution makes the classifier strongly biased towards the head classes. Therefore, we further propose two strategies to refine

the classifier under class-balanced sampling in the second stage while freezing the backbone. First, we add a learnable weight scaling (LWS) module to adjust the classifier to respect tail classes. Second, we integrate the KL-divergence loss to keep noticing head classes through knowledge distillation from the first stage. Coupling these two designs enables us to train an unbiased DNN model on both tail and head classes in oracle character recognition, thus offering the strong potential to improve both the average and total accuracies. We evaluate the proposed method on benchmarks including OBC306, Oracle-AYNU, and Oracle-20K. Experimental results show that our novel design attains new state-of-the-art performance.

## 2 Related Work

### 2.1 Oracle Character Recognition

Identifying characters from hand-copied or scanned oracle bone images has long been considered as a challenging problem. It has attracted much attention and achieved tremendous advances [1,2,18]. Earlier studies often adopted traditional pattern recognition techniques on oracle character recognition. For example, the work in [18] treated oracle bone inscriptions as undirected graphs and applied graph isomorphism for identification. Guo et al. [1] proposed a hierarchical representation for oracle characters, consisting of a Gabor-related low-level representation and a sparse-encoder-related mid-level representation. Liu et al. [19] recognized oracle characters by extracting block histogram-based features and employing support vector machines.

Recently, DNN-based methods have also been applied in oracle character recognition. In the early stage, researchers combined Convolutional Neural Network (CNN) models with traditional feature representation [1]. Next, Huang et al. [2] evaluated several popular CNNs (e.g., ResNet, InceptionNet) in their established OBC306. As DNN models usually require a large number of labelled samples for training, researchers have to make significant efforts on the oracle data collection. To this end, Guo et al. [1] first collected about 20,000 legible oracle character images called Oracle-20k. Then, Anyang Normal University constructed another hand-copied dataset Oracle-AYNU [1], and Huang et al. [2] released a large-scale scanned oracle character dataset called OBC306.

Owing to the difficulty in obtaining oracle characters, current oracle data is both rare and seriously long-tailed, making the DNN-based recognition of oracle characters challenging. Zhang et al. [3] first investigated the seriousness of this issue and proposed a nearest neighbour classifier with metric learning. Following that, Li et al. [5] integrated mix-up augmentation and triplet loss to improve the recognition performance. However, such long-tailed distribution issue is far from being solved. In this paper, we aim to train an unbiased DNN model for long-tailed oracle character recognition via the proposed two-stage decoupled learning method.

### 2.2 Long-Tailed Visual Recognition

It is crucial to obtain an unbiased model for all classes in long-tailed visual recognition. Most existing works can be divided into two categories: re-sampling and re-weighting. Re-sampling-based techniques typically obtain a more balanced data distribution by over-sampling tail classes or under-sampling head classes [6,7]. Re-weighting methods assign appropriate weights to the loss of each class to re-balance classes [8,9]. In addition, some methods adjust model logits during training or post-processing based on label frequencies to achieve relatively large margins between classes, which can also strengthen the classification of tail classes and mitigate the long-tailed distribution issue [10,11].

Recently, researchers have started studying two-stage-based decoupled learning in DNN models for imbalanced recognition instead of end-to-end learning. Kang et al. [12] proposed to decouple the one-stage training of the DNN model into feature representation (i.e., backbone network) and classifier learning. This work demonstrated that it could learn well-generalized representation under the normal instance-balanced sampling in the first training stage, and merely adjusting the classifier in the second stage is effective for imbalanced recognition. Based on this study, researchers innovated the decoupled learning scheme from different aspects. For example, KCL [13] developed a k-positive contrastive loss to learn a more balanced and discriminative feature representation. MiSLAS [14] proposed to adopt mixup augmentation in the first stage to enhance the representation learning and applied a label-aware smoothing strategy in the second stage. The work [15], following the weight re-balancing direction, proposed to tune weight decay in the first stage and utilized class-balanced loss with tuning weight decay in the second stage.

Despite the effectiveness, these present approaches conduct their evaluation on the test data, which usually follows a balanced distribution across all classes. In case of imbalanced test data, these methods would largely degrade their performance as they aim to obtain a uniform distribution during the training. Several works have noticed such an issue which is still less explored [20,21]. In this paper, we focus on decoupled learning for long-tailed oracle character recognition with imbalanced test data.

### 2.3 Knowledge Distillation

Knowledge distillation (KD) is proposed to achieve better generalization performance by transferring knowledge from pre-trained models (i.e., teachers) to target networks (i.e., students) [22]. The concept of KD was proposed by Hinton et al. [23], which transfers the knowledge via minimizing the KL-Divergence loss between predicted logits of teachers and students. Several recent works have explored KD for imbalanced visual recognition. Xiang et al. [24] divided the whole dataset into subsets and trained multiple teacher models for each subset. Meanwhile, a unified student model was trained by using adaptive KD in an easy-to-hard curriculum instance selection approach. Following this multi-expert framework, Wang et al. [20] introduced one special KD approach to simplify the

multi-expert model by training a student network with multiple experts. In this paper, we propose to integrate the idea of knowledge distillation in our two-stage-based decoupled learning framework for long-tailed oracle recognition, aiming to make the classifier not ignore the head classes by transferring the knowledge from the first stage.

## 3 Main Methodology

In this paper, we develop a two-stage decoupled learning to train an unbiased DNN model for both tail and head classes in long-tailed oracle character recognition. The overview of our work is illustrated in Fig. 3. In the first stage, the DNN model is trained by the cross-entropy loss under instance-balanced sampling. To be noted, we divide the DNN model into two components, i.e., backbone and classifier. Due to the insufficient oracle data, we adopt mixup augmentation to explore the training samples in the first stage fully. Although a well-generalized representation can be obtained from the backbone in the first stage [12], the classifier is usually biased to head classes because of the long-tailed distribution. To overcome this issue, we train a learnable weight scaling (LWS) module to adjust the classifier with the frozen backbone under class-balanced sampling in the second stage, thus paying more attention to the tail classes. In addition, we integrate the KL-Divergence loss in the second stage to maintain attention on the head classes by the knowledge distillation from the first stage. In the inference stage, the oracle character images pass through the backbone and classifier with LWS to output the final recognition result.

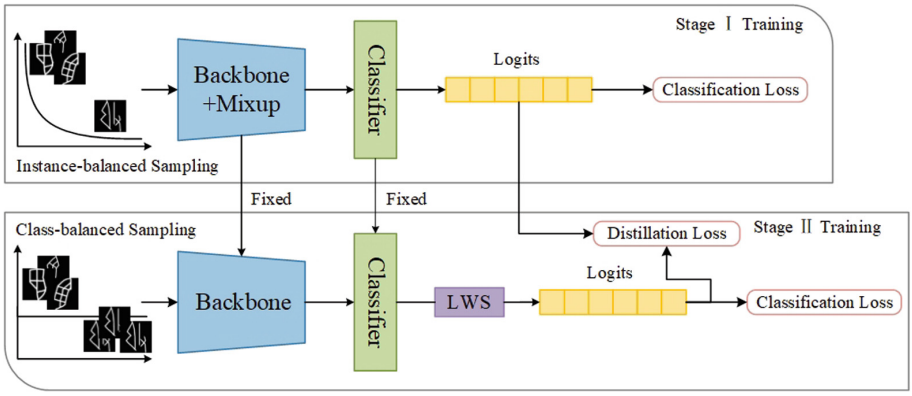



**Fig. 3.** Overall structure of the proposed scheme. LWS represents the learnable weight scaling, aiming to adjust the classifier to respect the tail classes.

### 3.1 Backbone Architecture

In terms of the excellent performance of Transformer in natural language processing and computer vision areas [16,25,26], we employ ViT [16] as the backbone network for oracle character recognition in this paper.

ViT mainly consists of patch embedding, position embedding, class token, and Transformer encoder [27]. First, we convert an image into a sequence of 1-D patch embeddings. Given an image $\boldsymbol{X} \in \mathbb{R}^{H\times W\times M}$ ($H, W, M$ denote the height, width, and the number of channels, respectively), we divide $\boldsymbol{X}$ into a series of patches denoted as $\boldsymbol{X}_p \in \mathbb{R}^{S\times P\times P\times M}$, where $P \times P$ is the patch size and $S$ is the number of patches (i.e., $S = HW/P^2$). Then, we flatten these cropped patches to be $\boldsymbol{X}_{p'} \in \mathbb{R}^{S\times(P^2\cdot M)}$ and utilize a trainable linear projection $\boldsymbol{E} \in \mathbb{R}^{(P^2\cdot M)\times D}$ to generate the sequence of 1-D embedding of all patches $\boldsymbol{X}_{p''} \in \mathbb{R}^{S\times D}$. Second, motivated from BERT [25], we add a learnable class token $\boldsymbol{x}_{class}$ to the beginning of the patch embeddings, which can be regarded as the representation of the input image and fed into the latter classifier. Third, we integrate the learnable position embedding $\boldsymbol{E}_{pos} \in \mathbb{R}^{(S+1)\times D}$ to retain the position information in the sequence as the Transformer encoder is permutation-invariant. Therefore, the overall input to the Transformer encoder can be defined as:

$$\boldsymbol{Z}_0 = [\boldsymbol{x}_{class}; \boldsymbol{X}_{p''}] + \boldsymbol{E}_{pos}. \tag{1}$$

The Transformer encoder is composed of $L$ layers. Every layer contains one multi-headed self-attention (MSA) block and one multi-layer perceptron (MLP) block. MLP includes two linear layers with a GELU. LayerNorm (LN) is applied before each block, while residual connections are applied after each block. More details can be seen in [27]. The process can be simply formulated as follows:

$$\begin{cases} \hat{\boldsymbol{Z}}_l = MSA(LN(\boldsymbol{Z}_{l-1})) + \boldsymbol{Z}_{l-1}, & l = 1 \ldots L \\ \tilde{\boldsymbol{Z}}_l = MLP(LN(\hat{\boldsymbol{Z}}_l)) + \hat{\boldsymbol{Z}}_l, & l = 1 \ldots L \\ \boldsymbol{z} = LN(\tilde{\boldsymbol{Z}}_L[0]). \end{cases} \tag{2}$$

### 3.2 Mixup Augmentation

Mixup [17] and its variants [28,29] have been widely adopted as data augmentation strategies in long-tailed tasks [5,14,30], which enable to improve the generalization of DNN models. The basic concept of Mixup is to generate new samples by interpolating two randomly sampled input images with their labels ($\boldsymbol{X}$, $\boldsymbol{y}$) and ($\boldsymbol{X}'$, $\boldsymbol{y}'$) as follows:

$$\tilde{\boldsymbol{X}} = \lambda \boldsymbol{X} + (1-\lambda)\boldsymbol{X}', \tag{3}$$

$$\tilde{\boldsymbol{y}} = \lambda \boldsymbol{y} + (1-\lambda)\boldsymbol{y}', \tag{4}$$

where $\tilde{\boldsymbol{X}}$ denotes the mixed new sample and its label is $\tilde{\boldsymbol{y}}$, $\lambda$ is a mixing factor drawn from a Beta distribution $Beta(\alpha, \beta)$ and $\alpha = \beta = 1$ in our experiments. We integrate this original Mixup augmentation method in the first training stage

for better representation learning of the backbone. Since the LWS module trained in the second stage is lightweight, we remove Mixup to reduce the complexity. Due to the limited space, we do not compare other variants of Mixup [28,29] in this paper, and more comparisons can be found in [5].

### 3.3 Decoupled Learning

In the paper, we decouple the DNN model into backbone and classifier in the two-stage training framework, where the backbone is learned in the first stage and the classifier is adjusted with the LWS module in the second stage for the long-tailed oracle character recognition.

In the first stage, we adopt the ViT model as the backbone to learn the representation $\boldsymbol{z}$ from the input $\boldsymbol{X}$, then obtain the classification logit $\hat{\boldsymbol{y}}$ by feeding $\boldsymbol{z}$ into the linear classifier. Finally, the predicted class could be given by argmax $\hat{\boldsymbol{y}}$ as follows:

$$\hat{\boldsymbol{y}} = C(\boldsymbol{z}) = \boldsymbol{W}^T z + \boldsymbol{b}, \tag{5}$$

where $C(\cdot)$ represents the classifier, $\boldsymbol{W}$ denotes the weight matrix and $\boldsymbol{b}$ denotes the bias. In this stage, the backbone and classifier are learned jointly by minimizing the standard cross-entropy loss between ground truth $y$ and argmax $\hat{\boldsymbol{y}}$ under instance-balanced sampling. Here, the probability of sampling data from the class $j$ is proportional to its cardinality $n_j$. Therefore, the long-tailed data distribution makes the learned model biased to head classes.

In the second stage, LWS aims to rectify the imbalanced decision boundaries between head and tail classes via re-scaling the magnitude of weights in $\boldsymbol{W}$ for each class. To this end, we utilize a scaling factor $f_j$ for the $j$-th class to adjust the weights of the classifier:

$$\tilde{\boldsymbol{w}_j} = f_j * \boldsymbol{w}_j, \tag{6}$$

where $\boldsymbol{w}_j \in \mathbb{R}^d$ represents the weight vector of class $j$. Then the whole weight matrix $\boldsymbol{W} = \{\boldsymbol{w}_j\} \in \mathbb{R}^{d \times Y}$ is re-scaled to $\tilde{\boldsymbol{W}}$ so that $\hat{\boldsymbol{y}}$ becomes $\tilde{\boldsymbol{W}}^T z + \boldsymbol{b}$, where $Y$ is the number of classes. In this way, merely the LWS block (denoted as $\boldsymbol{f} = \{f_j\} \in \mathbb{R}^Y$) is learned by class-balanced sampling in the second stage, while the backbone and classifier (i.e., $\boldsymbol{W}$ and $\boldsymbol{b}$) are fixed. Under class-balanced sampling, each class shares an equal probability of being selected. Once a class is selected, an instance is sampled uniformly from the chosen class, so it is unbiased sampling. To be noted, LWS is very lightweight, thereby its learning can converge quickly in the second stage of training.

### 3.4 Logit-Based Knowledge Distillation

Although LWS in the second stage can promote the learned DNN model to highlight tail classes, it will lose the importance of the head classes. However, the oracle test data is also a long-tailed distribution, and the recognition of head classes plays a crucial role in the overall performance. Since the model learned in the first stage better understands the head classes, we can utilize this model to

guide LWS in the second stage to keep attention on head classes. Motivated by this, we propose a knowledge distillation strategy in the second training stage.

Logit-based knowledge distillation aims to transfer the knowledge from a teacher model to a student model by aligning their logit predictions [22]. In this paper, we leverage the first stage model as the teacher model to guide the second stage model, where the popular soft targets [23] are adopted as our logit-based knowledge. Specifically, given classification logits $\hat{\boldsymbol{y}}$ of the classes, the soft targets can be obtained by the softmax function as follows:

$$p_j = \frac{\exp(\hat{y}_j/T)}{\sum_{t=1}^{Y}\exp(\hat{y}_t/T)}, \tag{7}$$

where $T$ denotes the temperature that controls the importance of each soft target and is set to 1 in our experiments. The classical KD adopts KL-Divergence as the distillation loss for soft targets, which can be rewritten as:

$$\mathrm{KL}(\boldsymbol{p}^T||\boldsymbol{p}^S) = \sum_{i=1}^{Y} p_i^T \log(\frac{p_i^T}{p_i^S}). \tag{8}$$

Here, $p^T$ and $p^S$ represent the teacher and student model output logits, respectively. By integrating the distillation loss in the second stage, the decision boundaries of head classes can be protected to some extent.

### 3.5 Overall Training

Our work follows a two-stage training scheme, which can be summarized as (1) representation learning by training the backbone and classifier under the cross-entropy loss with instance-balance sampling, and (2) classifier learning by training the integrated LWS under the cross-entropy and KL losses with class-balanced sampling.

In the first training stage, we aim to obtain well-generalized representations and achieve higher performance in the head classes for subsequent knowledge transfer. According to [12], instance-balanced sampling with the cross-entropy loss yields a more general representation than other re-sampling methods. Therefore, we adopt this training strategy in the paper. In addition, we integrate Mixup [17] to improve further the representation ability and recognition performance motivated by the previous works [5,14]. The loss function in the first stage is the cross-entropy loss:

$$L_{s1} = \sum_{n=1}^{N} -\tilde{\boldsymbol{y}}_n \log(C(B(\tilde{\boldsymbol{X}}_n))), \tag{9}$$

where $N$ is the number of training samples. $B(\cdot)$ represents the backbone network.

In the second stage, Zhong et al. [14] indicate that Mixup has negligible or even negative effects on classifier learning. Moreover, our trainable LWS is

lightweight. Therefore, we remove Mixup at this stage. Furthermore, to keep noticing head classes, the distillation loss is adopted for classes with larger cardinalities. The overall loss function of the second stage is defined as:

$$L_{s2} = \sum_{n=1}^{N} -\boldsymbol{y}_n \log(C(B(\boldsymbol{X}_n))) + \sum_{i \in \mathbb{N}} \mathrm{KL}_i(\boldsymbol{p}^T || \boldsymbol{p}^S). \tag{10}$$

Here, $\mathbb{N}$ represents the subset of classes whose cardinality is larger than $\gamma$, indicating those classes with knowledge distillation. In our experiments, we set $\gamma$ as 100, 150 and 160 in OBC306, Oracle-AYNU and Oralce-20k, respectively, which are tuned empirically.

## 4 Experiments

### 4.1 Datasets

**OBC306.** OBC306 [2] is currently the largest public dataset of oracle bone scripts to our knowledge, which contains 309,551 character samples with 306 classes in total. As shown in Fig. 2, this dataset suffers from a typical long-tailed distribution, i.e., 70 classes cover 83.82% of total samples while 52 classes have fewer than ten samples. We remove 29 classes with only one sample since we do not consider the out-of-vocabulary (OOV) performance. Then, the remaining dataset is divided into training and test sets following a 3:1 ratio. Finally, OBC306 used in the paper has 277 classes with 309,522 samples. The imbalance ratios (i.e., the size of the largest class: the smallest class) of the training set and test set are 19,424:1 and 6,474:1, respectively. To be noted, all samples in OBC306 are oracle bone rubbing images with various noises.

**Oracle-AYNU.** Oracle-AYNU [3] consists of 39,072 hand-copied oracle character samples with 2,584 classes. Specifically, the cardinality of each class varies from 2 to 287, and about 68% of classes have fewer than ten samples. We divide the dataset into the training and test sets following a 9:1 ratio, and then the imbalance ratios of the training set and test are 259:1 and 28:1, respectively. We can see that this dataset also suffers a long-tailed distribution issue, but not severe as that in OBC306.

**Oracle-20K.** Oracle-20K [1] contains 19,491 hand-copied samples with 249 classes, where class cardinalities range from 25 to 291. We split the dataset into training and test sets following a ratio of 2:1, and then the imbalance ratio of the training set and test set are 194:17 and 97:8, respectively. We can see that its imbalanced issue is not very severe compared to the other two benchmark datasets.

### 4.2 Implementation Details

We implement our model by Pytorch. In all experiments, we adopt ViT-Base [16] pre-trained on ImageNet as the backbone model, and utilize the SGD optimizer

with momentum 0.9, batch size 64, image size 256 × 256. In the first stage, the initial learning rate is 0.01 and deceased by 0.1 at the $m_1$-th and $m_2$-th epochs (i.e., $m_1 = 15$ and $m_2 = 20$ in OBC306, $m_1 = 100$ and $m_2 = 150$ in Oracle-AYNU, $m_1 = 150$ and $m_2 = 200$ in Oracle-20K). In the second stage, we restart the learning rate to train the LWS module with 0.2 for OBC306 and 0.01 for both Oracle-AYNU and Oracle-20K. The learning rate is decreased by 0.1 at $m'_1$-th and $m'_2$-th epochs (i.e., $m'_1 = 2$ and $m'_2 = 4$ in both OBC306 and Oracle-AYNU, $m'_1 = 5$ and $m'_2 = 10$ in Oracle-20K).

### 4.3 Evaluation Metrics

Most previous papers adopt the total accuracy as defined in Eq. (11) to evaluate the recognition performance. However, if the test data also follows long-tailed data distribution, this metric will be dominated by those head classes. To reflect the effectiveness on tail classes as well, we exploit another metric additionally to evaluate average accuracy as defined in Eq. (12). Following [5], such two metrics are formulated by

$$Total = \frac{\sum_{j=1}^{Y} r_j}{\sum_{j=1}^{Y} n_j}, \tag{11}$$

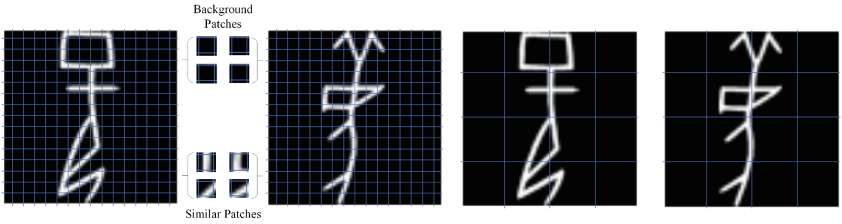
$$Average = \frac{1}{Y} \sum_{j=1}^{Y} \frac{r_j}{n_j}, \tag{12}$$

where $r_j$ and $n_j$ denote the number of correctly classified samples and total samples in the $j$-th class, respectively. In addition, to better demonstrate the performance on the long-tailed distribution, we split each dataset into three subsets, namely $Head$ (more than 100 samples), $Medium$ (20∼100 samples), and $Tail$ (fewer than 20 samples).

### 4.4 Ablation Study

**Effects of different patch sizes of ViT.** Similar to ViT [16], we also compare the effectiveness of different patch sizes in oracle character recognition. As shown in Fig. 4, if the patch size is too small, there will be too many less expressive patches input to the Transformer encoder, e.g., background patches or similar patches; if the patch size is too large, some non-trivial spatial and local information will be lost, leading to low discriminability. We try three sizes without pretraining, including 16 × 16, 32 × 32 and 64 × 64. The results are reported in Table 1. We can see that 32 × 32 provides the best performance. Furthermore, ViT usually requires pre-training on a large amount of data and then transferring to small datasets to obtain good results. Therefore, we choose the "Base" variant of ViT with 32 × 32 patch size pre-trained on ImageNet as our backbone, and fine-tune it together with the linear classifier on oracle datasets.

**Effects of Different Components.** In this part, we evaluate the effectiveness of the proposed LWS and KD. For clarity, we also report the performance of




(a) Small input patches (b) Large input patches

**Fig. 4.** Visualization of different patch sizes.

**Table 1.** Comparison on OBC306, Oracle-AYNU and Oracle-20K in terms of the average and total accuracies (%) of ViT-Base with different patch sizes.

| Patch Size | OBC306 | | Oracle-AYNU | | Oracle-20K | |
|---|---|---|---|---|---|---|
| | Average | Total | Average | Total | Average | Total |
| $16 \times 16$ | 65.36 | 87.81 | 61.54 | 73.84 | 82.78 | 86.79 |
| $32 \times 32$ | **68.08** | **88.97** | **66.65** | **77.88** | **87.80** | **90.76** |
| $64 \times 64$ | 60.38 | 85.88 | 65.18 | 76.68 | 86.30 | 90.20 |

Mixup. The results are summarized in Table 2, and its first row shows the baseline model where the backbone and classifier are jointly trained in one stage without any proposed component. Compared with the baseline model, we can see that Mixup (the second row in Table 2) improves the performance significantly in terms of both the average and total accuracy, demonstrating the effectiveness of the Mixup strategy.

To facilitate the recognition of tail classes, we learn LWS in the second stage of training to adjust the decision boundaries from the baseline model as shown in the third row of Table 2. We observe that LWS improves the average accuracy, especially on OBC306 and Oracle-AYNU, with significant gains of 5.08% and 4.79%, respectively, since both datasets have many tail classes. LWS adjusts the weights of the classifier under class-balanced sampling, which favors tail classes while weakening the dominance of head classes. Therefore, we suppose that the learned LWS will reduce the performance of the head classes, resulting in a reduction of total accuracy on OBC306. As OBC306 has an extremely large imbalance ratio (i.e., 9,424:1 and 6,474:1 in the training and test set, respectively) and the head classes dominate the test set, a slight suppression of the head classes significantly impacts the total accuracy. In contrast, the total accuracy is improved on Oracle-AYNU as the tail classes account for a large proportion in this dataset; however, the improvement is fairly smaller than the average accuracy. Combining Mixup and LWS (the fifth row of Table 2), recognition performance can be lifted up further, but the total accuracy on OBC306 is still reduced significantly.

To mitigate the issue of LWS on head classes, we plug the distillation loss in the second training stage to preserve the decision boundaries of head classes. The

**Table 2.** Ablation study for three proposed components on OBC306, Oracle-AYNU, and Oracle-20K in terms of the average and total accuracies (%). The first row is the baseline model. Mixup: adding Mixup to the baseline model. LWS: training LWS in the second stage. KD: adding the distillation loss in the second stage.

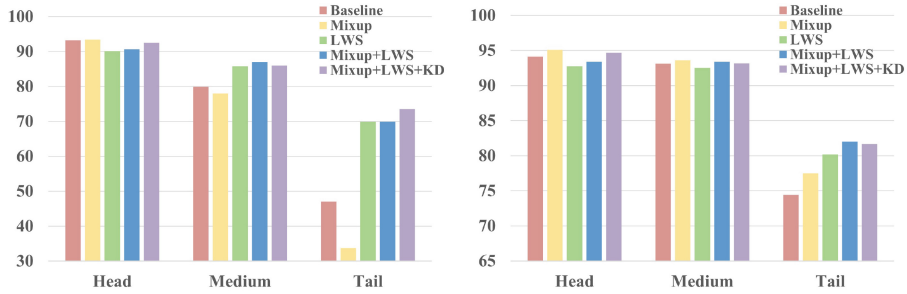
| Mixup | LWS | KD | OBC306 | | Oracle-AYNU | | Oracle-20K | |
|---|---|---|---|---|---|---|---|---|
| | | | Average | Total | Average | Total | Average | Total |
| ✗ | ✗ | ✗ | 79.50 | 93.02 | 77.07 | 84.42 | 93.78 | 95.37 |
| ✔ | ✗ | ✗ | 77.07 | **93.12** | 79.75 | 86.23 | 94.57 | 96.03 |
| ✗ | ✔ | ✗ | 84.58 | 90.05 | 81.86 | 86.68 | 93.92 | 95.32 |
| ✔ | ✔ | ✗ | 84.38 | 90.55 | **83.53** | 87.97 | 94.87 | 96.03 |
| ✔ | ✔ | ✔ | **85.01** | 92.40 | 83.27 | **88.02** | **94.9** | **96.06** |

results are listed in the last row of Table 2. On the one hand, adding KD boosts the total accuracy, especially on OBC306, from 90.55% to 92.40%; on the other hand, it makes little impact on the average accuracy, with a slight increase on OBC306 and Oracle-20K due to its effectiveness on the head classes. In summary, by combining all the proposed components, our proposed method can achieve a good trade-off between total and average accuracies, finally improving the overall recognition performance significantly.

To demonstrate the effectiveness of each component in detail, we show the total accuracy for the three subsets of OBC306 and Oracle-AYNU in Fig. 5. In OBC306, we can see that Mixup benefits the head subset more due to the severe long-tailed problem; after adding LWS, the accuracies of the tail and medium classes boost while head accuracy drops significantly by 2.76%. To keep noticing head classes, we adopt knowledge distillation to keep the decision boundaries of head classes while facilitating the tail classes. The proposed method (Mixup+LWS+KD) obtains a fair trade-off between the head and tail classes. Finally, it increases the tail accuracy by 3.61% compared to Mixup+LWS, without much loss in head performance compared to Mixup. In Oracle-AYNU, Mixup generates positive influences on all the three subsets. For LWS, it improves the accuracy of tail classes while reducing the accuracy of head classes. By combining it with KD, the accuracies of all the three subsets increase compared with the baseline model, from 94.14% to 94.68%, 93.12% to 93.19% and 74.41% to 81.69%, respectively. In summary, these results demonstrate the effectiveness of our proposed method apparently.

### 4.5 Comparison to Previous Methods

In Table 3, we compare the proposed method with the previous competitive oracle character recognition methods regarding average and total accuracy on three benchmark datasets. From the table, we can see that our approach surpasses the other methods on both average and total accuracy in all datasets. Specifically, the proposed method achieves superior performance on OBC306 with the serious long-tailed data issue, improving the average and total accuracies over




(a) OBC306 (b) Oracle-AYNU

**Fig. 5.** Total accuracy of each component on the head, medium and tail classes of OBC306 and Oracle-AYNU.

**Table 3.** Comparison to previous methods on Oracle-AYNU, OBC306, and Oracle-20K. † denotes the accuracy excluding 29 classes which do not have any training samples (The original average accuracy is reported as 70.28% on all the 306 classes [2]).

| Method | OBC306 | | Oracle-AYNU | | Oracle-20K | |
|---|---|---|---|---|---|---|
| | Average | Total | Average | Total | Average | Total |
| Zhang et al. [3] | – | – | – | 83.37 | – | 92.43 |
| Huang et al. [2]† | 77.64 | – | – | – | – | – |
| Li et al. [5] | 80.16 | 91.74 | 79.96 | 83.65 | 93.50 | 94.64 |
| Ours | **85.01** | **92.4** | **83.27** | **88.02** | **94.9** | **96.06** |

the previous best oracle character recognition method [5] by 4.85% and 0.66%, respectively. For the Oracle-AYNU dataset with more than 2,000 classes, the proposed method achieves 83.27% in average accuracy and 88.02% in total accuracy, which outperforms the prior methods by a significant margin of 3.31% and 4.37%, respectively. In Oracle-20K, the long-tailed issue is not evident; however, our proposed method can also yield better average and total results. In summary, the proposed method achieves new state-of-the-art performance on three benchmark datasets.

### 4.6 Error Analyses

In Fig. 6, we present some error recognition examples of the proposed approach, which are roughly summarized into four categories: long tail, similar looking, intra-class variance, and severe noises. First, the long-tailed issue has not been fully resolved, and some tail characters are mis-recognized as shown in Fig. 6(a). Second, it is sometimes confusing for our model to identify similar characters. For example, the characters in Fig. 6(b) look very similar between the ground truth and prediction (e.g., in the second row, the character '1035' is mis-recognized to '100003'). Third, since there was no standardization of the oracle bone script during historical periods, some characters may have a high degree of intra-class

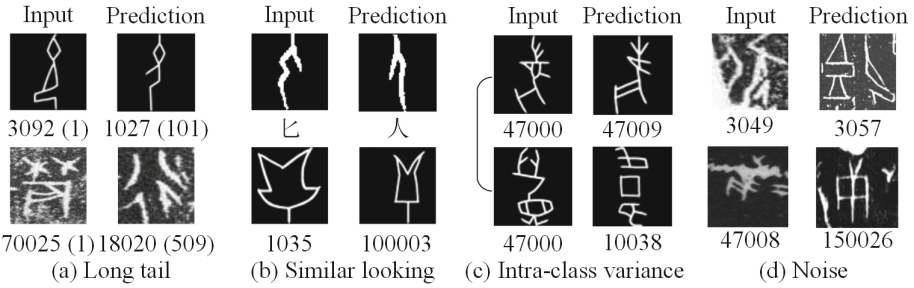



**Fig. 6.** Error examples of our model. Input columns present the input images with their ground-truth labels underneath (characters or digital codes). Prediction columns present samples from the corresponding incorrect predicted classes. Note that Oracle-AYNU and OBC306 are annotated by digital codes, not by real characters like Oracle-20K. In (a), the number in parentheses represents the number of training samples for that class. In (c), round brackets represent they belong to the same class.

variation in shapes, structures, and number of strokes in Fig. 6(c). Last, due to the long history, the real oracle characters on bones are polluted by serious noises and abrasions as shown in Fig. 6(d), making them difficult to be recognized even by humans.

## 5 Conclusion

In this paper, we propose a two-stage decoupled learning method for long-tailed oracle character recognition, aiming to train an unbiased DNN model on both tail and head classes. In the first stage, we train a ViT backbone model and a linear classifier under the instance-balanced sampling, where mixup augmentation is utilized to exploit current oracle samples fully. In the second stage, we propose a learnable weight scaling module to refine the classifier to respect tail classes. Meanwhile, the KL-divergence loss is also integrated to maintain attention on head classes by the knowledge distillation from the first stage. Extensive experiments on three oracle benchmark datasets demonstrate the effectiveness of both LWS and KD components, finally achieving new state-of-the-art recognition performance on average and total accuracy. However, there is still room to improve further the recognition performance, especially on tail classes. In the future, we will explore how to enhance the representation ability of the backbone on tail classes so as to further promote the decoupled learning framework.

**Acknowledgements.** This research was funded by National Natural Science Foundation of China (NSFC) no.62276258, Jiangsu Science and Technology Programme (Natural Science Foundation of Jiangsu Province) no. BE2020006-4, and Xi'an Jiaotong-Liverpool University's Key Program Special Fund no. KSF-T-06.

## References


1. Guo, J., Wang, C., Roman-Rangel, E., et al.: Building hierarchical representations for oracle character and sketch recognition. IEEE Trans. Image Process. **1**, 104–118 (2016)
2. Huang, S., Wang, H., Liu, Y., et al.: OBC306: a large-scale oracle bone character recognition dataset. In: Proceedings of International Conference on Document Analysis and Recognition (ICDAR), pp. 681–688 (2019)
3. Zhang, Y.-K., Zhang, H., Liu, Y.-G., et al.: Oracle character recognition by nearest neighbor classification with deep metric learning. In: Proceedings of International Conference on Document Analysis and Recognition (ICDAR), pp. 309–314 (2019)
4. Huang, K., Hussain, A., Wang, Q.-F., Zhang, R.: Deep Learning: Fundamentals, Theory and Applications, vol. 2 (2019)
5. Li, J., Wang, Q.-F., Zhang, R., Huang, K.: Mix-up augmentation for oracle character recognition with imbalanced data distribution. In: Proceedings of International Conference on Document Analysis and Recognition (ICDAR), pp. 237–251 (2021)
6. 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)
7. Estabrooks, A., Jo, T., Japkowicz, N.: A multiple resampling method for learning from imbalanced data sets. Comput. Intell. **20**(1), 18–36 (2004)
8. Cui, Y., Jia, M., Lin, T.-Y., et al.: Class-balanced loss based on effective number of samples. In: Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR), pp. 9268–9277 (2019)
9. Cao, K., Wei, C., Gaidon, A., et al.: Learning imbalanced datasets with label-distribution-aware margin loss. In: Proceedings of Advances in Neural Information Processing Systems (NeurIPS), pp. 1565–1576 (2019)
10. Menon, A.K., Jayasumana, S., Rawat, A.S., et al.: Long-tail learning via logit adjustment. In: Proceedings of International Conference on Learning Representations (ICLR) (2021)
11. Wu, T., Liu, Z., Huang, Q., et al.: Adversarial robustness under long-tailed distribution. In: Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8659–8668 (2021)
12. Kang, B., Xie, S., Rohrbach,, M., et al.: Decoupling representation and classifier for long-tailed recognition. In: Proceedings of International Conference on Learning Representations (ICLR) (2020)
13. Kang, B., Li, Y., Xie, S., et al.: Exploring balanced feature spaces for representation learning. In: Proceedings of International Conference on Learning Representations (ICLR) (2021)
14. Zhong, Z., Cui, J., Liu, S., Jia, J.: Improving calibration for long-tailed recognition. In: Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR), pp. 16489–16498 (2021)
15. Alshammari, S., Wang, Y.-X., Ramanan, D., Kong S.: Long-tailed recognition via weight balancing. In: Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6887–6897 (2022)
16. Dosovitskiy, A., Beyer, L., Kolesnikov, A., et al.: An image is worth 16x16 words: transformers for image recognition at scale. In: Proceedings of International Conference on Learning Representations (ICLR) (2021)
17. Zhang, H., Cissé, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: beyond empirical risk minimization. In: Proceedings of International Conference on Learning Representations (ICLR) (2018)

18. Li, Q., Yang, Y., Wang, A.: Recognition of inscriptions on bones or tortoise shells based on graph isomorphism. Jisuanji Gongcheng yu Yingyong (Comput. Eng. Appl.) **47**(8), 112–114 (2011)
19. Liu, Y., Liu, G.: Oracle bone inscription recognition based on SVM. J. Anyang Normal Univ. **2**, 54–56 (2017)
20. Wang, X., Lian, L., Miao, Z., et al.: Long-tailed recognition by routing diverse distribution-aware experts. In: Proceedings of International Conference on Learning Representations (ICLR) (2021)
21. Zhang, Y., Hooi, B., Hong, L., Feng, J.: Self-supervised aggregation of diverse experts for test-agnostic long-tailed recognition. In: Advances in Neural Information Processing Systems (NeurIPS) (2022)
22. Gou, J., Baosheng, Yu., Maybank, S.J., Tao, D.: Knowledge distillation: a survey. Int. J. Comput. Vision **129**(6), 1789–1819 (2021)
23. Hinton, G.E., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network (2015)
24. Xiang, L., Ding, G., Han, J.: Learning from multiple experts: self-paced knowledge distillation for long-tailed classification. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12350, pp. 247–263. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58558-7_15
25. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), pp. 4171–4186 (2019)
26. He, K., Chen, X., Xie, S., et al.: Masked autoencoders are scalable vision learners. In: Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR), pp. 15979–15988 (2022)
27. Vaswani, A., Shazeer, N., Parmar, N., et al.: Attention is all you need. In: Proceedings of Advances in Neural Information Processing Systems (NeurIPS), pp. 5998–6008 (2017)
28. Verma, V., Lamb, A., Beckham, C., et al.: Manifold mixup: better representations by interpolating hidden states. In: Proceedings of International Conference on Machine Learning (ICML), pp. 6438–6447 (2019)
29. Yun, S., Han, D., Chun, S., et al.: Cutmix: regularization strategy to train strong classifiers with localizable features. In: Proceedings of International Conference on Computer Vision (ICCV), pp. 6022–6031 (2019)
30. Zhang, Y., Wei, X.-S., Zhou, B., Wu, J.: Bag of tricks for long-tailed visual recognition with deep convolutional neural networks. In: Proceedings of Association for the Advancement of Artificial Intelligence (AAAI), pp. 3447–3455 (2021)