Distilling Virtual Examples for Long-tailed Recognition

Yin-Yin He[1], Jianxin Wu[1]*, Xiu-Shen Wei[2,1]

[1]State Key Laboratory for Novel Software Technology, Nanjing University, China

[2]School of Computer Science and Engineering, Nanjing University of Science and Technology, China

heyy@lamda.nju.edu.cn, {wujx2001, weixs.gm}@gmail.com

Abstract

We tackle the long-tailed visual recognition problem from the knowledge distillation perspective by proposing a Distill the Virtual Examples (DiVE) method. Specifically, by treating the predictions of a teacher model as virtual examples, we prove that distilling from these virtual examples is equivalent to label distribution learning under certain constraints. We show that when the virtual example distribution becomes flatter than the original input distribution, the under-represented tail classes will receive significant improvements, which is crucial in long-tailed recognition. The proposed DiVE method can explicitly tune the virtual example distribution to become flat. Extensive experiments on three benchmark datasets, including the large-scale iNaturalist ones, justify that the proposed DiVE method can significantly outperform state-of-the-art methods. Furthermore, additional analyses and experiments verify the virtual example interpretation, and demonstrate the effectiveness of tailored designs in DiVE for long-tailed problems.

1. Introduction

Deep convolutional neural networks have achieved remarkable success in various fields of computer vision, part of which should be attributed to rich and representative datasets. Manually-built datasets are often well-designed and roughly balanced, with sufficient samples for every category, *e.g.*, ImageNet ILSVRC 2012 [22]. In the real world, however, image data are often inherently long-tailed. A few categories (the "head" categories) contain most training images, while most categories (the "tail" ones) have only few samples. Some recently released datasets start to draw our attention to this practical setting, *e.g.*, iNaturalist [4] and LVIS [8]. These datasets show a naturally long-tailed dis-

tribution. Models trained on them are easily biased towards the head classes, while the tail categories often have much lower accuracy rates compared to the head ones. This bias is, of course, not welcome by researchers or practitioners.

Many attempts have been made to deal with long-tailed recognition [16, 3, 1, 34, 15, 14]. In particular, resampling makes a more balanced distribution through under-sampling [10, 5] the head classes or oversampling the tail classes [2, 9, 23]. Another direction, reweighting, is to assign higher costs for tail categories in novel loss functions [3, 25, 16]. Recent methods [34, 15] also decouple the training of the backbone network and the classifier part. These methods, however, *never cross the category boundary*. That is, resampling, reweighting, and decoupling all *happen independently inside each category*, and there is *no interaction among different categories*.

A simple but interesting experiment motivated us to utilize cross-category interactions for long-tailed recognition. The complete CIFAR-100 dataset is balanced, and a highly imbalanced subset of it (*i.e.*, CIFAR-100-LT) is a widely used benchmark for long-tailed recognition [34]. We use the entire CIFAR-100 training set to train a (teacher) network, and then use knowledge distillation [12] to distill a student network on the long-tailed CIFAR-100-LT with imbalance factor 100. The student's test accuracy is 61.58%, which is significantly (more than 10 percentage points) higher than existing long-tail recognition methods (*c.f.* Table 1)! Then, what makes its accuracy so high aside from the teacher being trained using the entire training set (which is not available in our long-tailed setting for the student)? Our answer to this question is two-fold: virtual examples & knowledge distillation, or in short, *distilling the virtual examples*.

In a dog vs. cat binary recognition problem, if the prediction for a dog image is $(0.7, 0.3)$, we interpret this prediction as two virtual examples: 0.7 dog virtual example, plus 0.3 cat virtual example. This interpretation extends naturally to the multiclass case. If dog is a head category and cat is a tail category, *the 0.3 cat virtual example will help recognize cats, even if the input image is in fact a dog*.
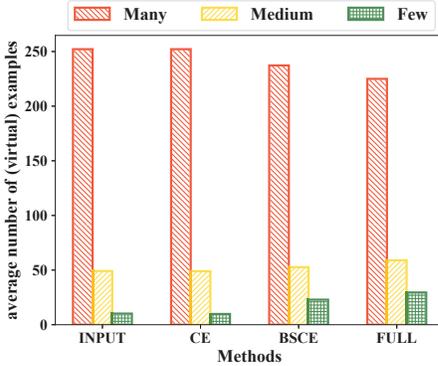

Figure 1. (Virtual) example distribution of different models.

Given a training set and a CNN model, we can compute the model's virtual example distribution on the training set by summing the contribution of all training examples to all categories. Note that *through virtual examples, different categories naturally interact in* every *training example*!

Empirically, we often divide the categories in a long-tailed problem into three subsets based on the number of training images in a category: *Many* (or head), *Medium*, and *Few* (or tail). Fig. 1 shows the average number of examples and virtual examples for these subsets of 4 different cases. The first ("INPUT") is the distribution for original input images in CIFAR-100-LT. The rest are the *virtual* example distributions for 3 models: cross entropy ("CE", *i.e.*, regular CNN training without any long-tail-specific learning), "BSCE" [21] (a long-tailed recognition method), and "FULL" (trained using CIFAR-100, as described above). Using the 3 models (*i.e.*, their virtual example distributions in Fig. 1) as teachers, the three students' accuracy on CIFAR-100-LT are 39.20%, 43.25%, and 53.71%, respectively.[1] That is, the more balanced the teacher's virtual example distribution, the higher the student's accuracy is.

These observations inspired us to propose a DiVE (distilling virtual examples) method, which has the following properties and contributions:

- **Validity of the virtual example interpretation.** In Sec. 3.1 and 3.2, we show that the virtual example interpretation is valid, which then allows us to utilize direct and explicit cross-category interactions.
- **Necessity of a balanced virtual example distribution**. Comparing INPUT with CE, the virtual example distribution of CE is almost identical to the original example distribution INPUT. However, we prove in Sec. 3.3 that *the virtual example distribution must be flatter* so long as we want to remove the bias against tail categories. Comparing CE, BSCE and FULL, we indeed observe

---

[1]The temperature $\tau = 1$ is used in Fig. 1 and in these experiments. Setting $\tau = 3$ leads to a more balanced virtual example distribution, and the accuracy is 61.58% for "FULL". More details about the temperature and distillation will be provided in Sec. 3.1.

empirically that the flatter the teacher's virtual example distribution, the higher the student's accuracy is.

- **To level and to distill the virtual example distribution (DiVE).** Noticing that even FULL in Fig. 1 is still long-tailed, we propose methods to make the virtual example distribution balanced, and then distill from it, which directly and explicitly learns from the balanced virtual example distribution.

As validated by experiments, the proposed DiVE method outperforms existing long-tailed recognition methods by large margins in various long-tailed recognition datasets.

2. Related Work

Recently, long-tailed recognition has attracted lots of attention [1, 34, 3, 15, 26], including in recognition and detection [16, 25]. We will briefly review previous methods on long-tailed recognition and knowledge distillation.

Resampling/reweighting: One classic way to deal with long-tail distribution is data resampling. The idea is to make the class distribution more balanced. It includes oversampling for minority categories [2, 9, 23] and undersampling for majority categories [10, 5] or learning to sample [21]. However, resampling can cause problems in deep learning [2, 3], *e.g.*, oversampling may lead to overfitting, while undersampling limits the generalization ability of neural networks. Another commonly used method is to reweigh the loss function [18]. This series of methods assign minority category instances more costs which are always misclassified or not confident [13, 3, 1, 16]. And balanced softmax [21] is proposed to replace the standard softmax transformation. These methods, however, all sacrifice the accuracy of the head to compensate for the tail.

Decoupled training: Recent works show that decoupling the representation and classifier learning improves the performance on long-tailed datasets [15, 34] significantly. However, they did not take into account the under-represented features of tail categories, which confines their improvements only to the classifier.

Knowledge transfer: To transfer knowledge from head to tail categories is another branch of methods [28, 17, 35, 29]. Specifically, [28] designed a module to use the head classes to learn the parameters of tail classes through meta-learning. [17, 35] transfer knowledge from head to tail through complex memory banks. [34, 29, 27] ensembled multiple experts' knowledge. Some methods explored self- and semi-supervised learning [31], but required much longer training time or extra training data. They are usually complex and hard to generalize on different tasks.

In short, existing methods either lack a mechanism that let head and tail categories interact with each other, or are too complex to generalize and utilize well. The proposed DiVE method, on the contrary, is a simple pipeline that utilizes knowledge distillation to distill from virtual examples.

Within this process, examples from different categories naturally interact with each other (i.e., head helps tail).

Knowledge distillation: Knowledge distillation (KD) is a technique to transfer knowledge across different models [12], which is most popular in model compression. Since its inception in [12], knowledge distillation has attracted lots of attention [31, 32]. Recently, [29, 19] invited KD to long-tailed problems. However, in DiVE, the key difference is that we have a different starting point by making the teacher's virtual example distribution to be flat.

Some works try to explain the mechanism behind KD. Specifically, [33] treated KD as a learnable label smoother, while we provide another interpretation. We argue that knowledge distillation shares knowledge among different classes through virtual examples, which is very similar to deep label distribution learning (DLDL) [6].

3. Distilling the Virtual Examples

We call the proposed method DiVE (**Di**stilling **V**irtual **E**xamples), which has a relatively simple pipeline: A teacher model is first trained for the long-tailed task with any existing methods, then we use knowledge distillation to transfer knowledge from the teacher (the virtual examples) to a student model. The distilled student model is DiVE's output.

Since the proposed DiVE method hinges on distilling virtual examples, we first establish an equivalence between knowledge distillation and deep label distribution learning (Sec. 3.1), then explain in detail how the label distribution interpretation leads to virtual examples (Sec. 3.2) in our context, then why the virtual example distribution must be flat (Sec. 3.3), and finally how to generate a balanced virtual example distribution to distill in long-tailed tasks (Sec. 3.4).

3.1. KD ≈ DLDL

In a C-class classification problem, assume a training set of n examples $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)\}$, where $\boldsymbol{x}_i$ is the i-th training instance and $y_i \in \{1, 2, \ldots, C\}$ is its groundtruth label. The one-hot encoding can turn y_i into an equivalent vector $\boldsymbol{y}_i = (y_{i,1}, y_{i,2}, \ldots, y_{i,C}) \in \mathbb{R}^C$, with the k-th component of $\boldsymbol{y}_i$ defined by $y_{i,k} = 1$ if $k = y_i$, otherwise $y_{i,k} = 0$. With a slight abuse of notations, we denote y_k as the k-th component of $\boldsymbol{y}$ from now on.

For a given training example $\boldsymbol{x}$ and its corresponding one-hot label $\boldsymbol{y}$, suppose a teacher CNN model predicts $\boldsymbol{t} = (t_1, t_2, \ldots, t_C) \in \mathbb{R}^C$ for it, which is obtained by transforming the logits $\boldsymbol{z} \in \mathbb{R}^C$ by a softmax function, as

$$t_i = \frac{\exp(z_i)}{\sum_{k=1}^C \exp(z_k)} \,. \tag{1}$$

Knowledge distillation (KD) [12] then uses knowledge hidden in $\boldsymbol{t}$ to help train a student network (which often has smaller capacity than the teacher network).

We can similarly denote the student network's prediction as $\boldsymbol{s} = (s_1, s_2, \ldots, s_C)$. Then, the student's loss function is

$$L_{\text{KD}} = (1 - \alpha)L_{\text{CE}}(\boldsymbol{y}, \boldsymbol{s}) + \alpha L_{\text{KL}}(\boldsymbol{t}, \boldsymbol{s}) \,. \tag{2}$$

The first term is the usual cross entropy (CE) loss between groundtruth labels and student predictions:

$$L_{\text{CE}}(\boldsymbol{y}, \boldsymbol{s}) = -\sum_{k=1}^C y_k \log s_k \,. \tag{3}$$

The second term encourages the student predictions to mimic the teachers' via minimizing their Kullback-Leibler (KL) divergence,

$$L_{\text{KL}}(\boldsymbol{t}, \boldsymbol{s}) = \sum_{k=1}^C t_k \log \frac{t_k}{s_k} \,. \tag{4}$$

Note that a temperature parameter τ is often used in KD. When $\tau \neq 1$, we need to compute $\boldsymbol{t}^\tau$ as

$$t_i^\tau = \frac{\exp(z_i/\tau)}{\sum_{k=1}^C \exp(z_k/\tau)} \,, \tag{5}$$

and similarly change $\boldsymbol{s}$ to $\boldsymbol{s}^\tau$, and the second loss term becomes $\tau^2 L_{\text{KL}}(\boldsymbol{t}^\tau, \boldsymbol{s}^\tau)$. The hyperparameter $\alpha \in [0, 1]$ balances these two loss terms, which is between 0 and 1. For now, we temporarily assume that $\tau = 1$.

Note that $\boldsymbol{y}, \boldsymbol{t}$ and $\boldsymbol{s}$ are all discrete distributions, and we use $H(\cdot)$ to denote the entropy. Let us define

$$\tilde{\boldsymbol{t}} = (1 - \alpha)\boldsymbol{y} + \alpha\boldsymbol{t} \,, \tag{6}$$

and use the well-known fact that $L_{\text{CE}}(\boldsymbol{x}, \boldsymbol{y}) = L_{\text{KL}}(\boldsymbol{x}, \boldsymbol{y}) + H(\boldsymbol{x})$, then it is easy to derive that

$$L_{\text{KD}} = (1 - \alpha)L_{\text{CE}}(\boldsymbol{y}, \boldsymbol{s}) + \alpha L_{\text{KL}}(\boldsymbol{t}, \boldsymbol{s}) \tag{7}$$

$$= (1 - \alpha)L_{\text{CE}}(\boldsymbol{y}, \boldsymbol{s}) + \alpha L_{\text{CE}}(\boldsymbol{t}, \boldsymbol{s}) - \alpha H(\boldsymbol{t}) \tag{8}$$

$$= L_{\text{CE}}\big((1 - \alpha)\boldsymbol{y} + \alpha\boldsymbol{t}, \boldsymbol{s}\big) - \alpha H(\boldsymbol{t}) \tag{9}$$

$$= L_{\text{CE}}(\tilde{\boldsymbol{t}}, \boldsymbol{s}) - \alpha H(\boldsymbol{t}) \tag{10}$$

$$= L_{\text{KL}}(\tilde{\boldsymbol{t}}, \boldsymbol{s}) + H(\tilde{\boldsymbol{t}}) - \alpha H(\boldsymbol{t}) \,. \tag{11}$$

On one hand, because $\boldsymbol{y}, \boldsymbol{t}$ (and hence $\tilde{\boldsymbol{t}}$) have zero gradients with respect to the student model's parameters, we immediately notice that $L_{\text{KL}}(\tilde{\boldsymbol{t}}, \boldsymbol{s})$ (or $L_{\text{CE}}(\tilde{\boldsymbol{t}}, \boldsymbol{s})$) is *an equivalent loss function* for training the student model. On the other hand, $L_{\text{KL}}(\tilde{\boldsymbol{t}}, \boldsymbol{s})$ is, as we will discuss in the next subsection, exactly the loss function of a DLDL [6] model. Hence, we have proved that *when the temperature $\tau = 1$, knowledge distillation is equivalent to DLDL.*

3.2. From label distributions to virtual examples

The label distribution learning (LDL) [7] handles tasks where the groundtruth labels are in fact *uncertain*. For example, estimating the apparent age based on a facial image is difficult—two annotators may give different answers for the same picture, *e.g.*, 25 and 27 years old. Hence, when the groundtruth label is 25, instead of using a one-hot encoding for 25, LDL generates a "label distribution" y as its label, with y_{25} being the largest, and other labels around 25 have non-zeros values, too. For example, treating 0 to 100 as $C = 101$ classification labels, the LDL label may be: $y_{25} = 0.7$, $y_{24} = y_{26} = 0.1$, $y_{23} = y_{27} = 0.05$, and $y_k = 0$ if $k < 23$ or $k > 27$. Note that the LDL label y is a valid distribution: $\sum_k y_k = 1$ and $y_k \geq 0$. The DLDL (deep LDL) method [6] combines LDL with the deep learning paradigm, and uses a KL-based loss $L_{KL}(y, t)$ to calculate the loss for a training example when its prediction is t. Hence, when $\tau = 1$, knowledge distillation is equivalent to DLDL with the groundtruth label distribution being $\tilde{t}$.

In KD, although the groundtruth label is assumed to be *certain*, the teacher model's prediction may be *wrong*. For example, the groundtruth label is 7, but $\arg\max_k t_k \neq 7$ may hold true. KD can correct such errors using the L_{CE} term, because it forces the prediction to match the groundtruth label. In DLDL, it was coerced through $\tilde{t}$—now $\tilde{t}_7 = (1 - \alpha) \cdot 1 + \alpha t_7$, and so long as $\alpha \leq 0.5$ we know for sure $\arg\max_k \tilde{t}_k = 7$ because $\tilde{t}_7 \geq 0.5$.

DLDL argues that a facial image with age 24 is in fact useful for classifying 25-year-old faces, because faces with "nearby" ages must share similar visual characteristics. Although the "nearby" concept does *not* apply to more general and long-tailed recognition problems, we now show that the teacher model's prediction in fact creates *virtual examples* that help long-tailed recognition, as illustrated in Fig. 2.

In this illustrative example, there are many dog images, but very few cat ones, therefore it is a long-tailed problem. The teacher model's prediction scores for all categories are nonzero due to the property of softmax. Hence, the original input image is replaced by virtual examples in 5 categories: 0.7 dog, 0.02 car, 0.07 rabbit, 0.01 ship and 0.2 cat in a KD process. Since a dog image's prediction has non-negligible mass (0.2) for cat, it means there are similarities between these categories, and *the virtual cat example will be useful in learning the cat category*, even though it is a dog image.

Note that the distillation loss is $L_{CE}(\tilde{t}, s)$ (Eqn. 10), which equals $-\sum_{k=1}^{C} \tilde{t}_k \log s_k$. For category k, there are $\hat{t}_k$ virtual example(s), and the virtual example's loss is $-\log s_k$ if we collect virtual examples from all categories and perform a normal cross-entropy training. Hence, summing up the losses of all virtual examples, we obtain $-\sum_{k=1}^{C} \tilde{t}_k \log s_k$, or equivalent to the distillation loss $L_{CE}(\tilde{t}, s)$. Hence, the virtual example interpretation is valid. For one category, if we sum the count of virtual ex-
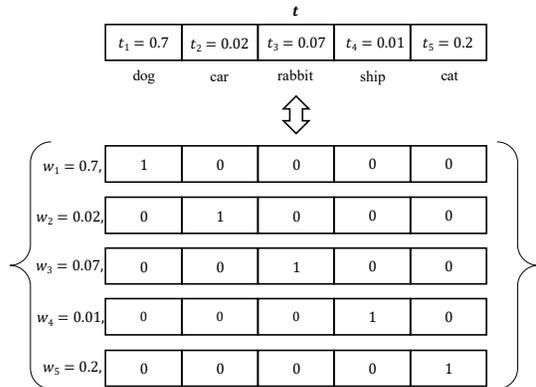

Figure 2. Illustration of virtual examples. An input dog example received prediction in the first row from the teacher model, and in effect creates virtual examples in all 5 categories: 0.7 dog example, 0.02 car example, 0.07 rabbit example, 0.01 ship example and 0.2 cat example.
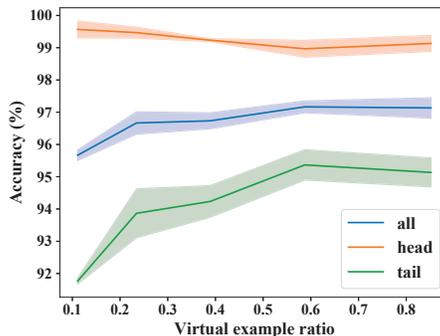

Figure 3. Accuracy (mean value and standard deviation) in a binary classification example. The accuracy becomes higher when the virtual example ratio between two classes grows (i.e., when the virtual example distribution becomes flatter). 'All' is the union of 'head' and 'tail'.

amples contributed to this category by all training examples, we obtain the number of virtual examples for this category.

It is worth noting that our Equations (6) and (11) are the same as those in [33]. However, their meanings and goals are different. [33] is motivated by label smoothing [24] and is assuming high temperature τ such that t is close to uniform for every example. However, in long-tailed problems, we only want the *distribution of all virtual examples* to become flatter, but every t must still carry useful information to discriminate different categories and transfer to s. In fact, we mostly use a small temperature (*e.g.*, $\tau = 1$ or $\tau = 3$).

3.3. The virtual example distribution must be flat

Now we further show that the virtual example distribution must be flat (or at least flatter than the original input image distribution).

We go back to the dog vs. cat binary recognition problem. What if the dataset is imbalanced? Suppose the dog has n_{head} samples while the cat has n_{tail} samples, and $n_{head} \gg n_{tail}$. We use label smoothing to generate virtual examples from dog for cat. Each dog sample is converted to ϵ virtual cat example and $1 - \epsilon$ dog example ($\epsilon < 0.5$). The number of virtual examples for cat and dog are $n_{tail} + n_{head} \times \epsilon$ and $n_{head} - n_{head} \times \epsilon$, respectively. We use the virtual example ratio between cat and dog to measure the flatness of the virtual example distribution.

This simple experiment was conducted on two classes randomly chosen from CIFAR-10, one with 5000 training samples, the other 500. We take "airplane" as head and "automobile" as tail, with results in Fig. 3. As the virtual example distribution goes flatter, performance of the tail increase significantly while the head is almost intact. The original label distribution learning requires correlations among different labels. But, in long-tailed recognition, Fig. 3 shows that *virtual examples from the head categories will help recognizing examples from the tail categories, even if these categories are* not *correlated*. For more examples, please refer to our supplementary materials.

Hence, we easily obtain the following conclusion: in order to obtain a balanced model for a long-tailed task, *the virtual example distribution must be much flatter than the input distribution*. The tail categories must have significantly more virtual examples than their number of input images, and the trend is reversed for the head categories. Otherwise, tail categories will have low accuracy rates. Fig. 1 clearly verifies this conclusion, in which "CE" is a failure case, while in "BSCE" and especially in "FULL", we observe better virtual example distributions.

Still, two difficulties exist. First, existing methods like BSCE often assign different weights to categories, but this kind of strategies have only limited effect on the virtual example distribution, because they can only affect the virtual example distribution in an indirect manner. Fig. 4 shows two such examples (BSCE [21] and LWS [15]). For example, BSCE [21] uses the following function to replace the softmax function in computing the soft logits s^{BSCE}:

$$s_i^{\text{BSCE}} = \frac{n_i \exp(z_i)}{\sum_{k=1}^{C} n_k \exp(z_k)}, \qquad (12)$$

where n_i is the number of training examples for category i. But, its virtual example distribution is still similar to that of the original input (*c.f.* Fig. 1). We need a *direct and explicit* way to obtain a flatter virtual example distribution.

Second, we still do not know what level of "flatness" is beneficial for long-tailed recognition.

3.4. Level and distill virtual examples

Our answers to addressing both difficulties are pretty straightforward: Distill the Virtual Examples (DiVE). In
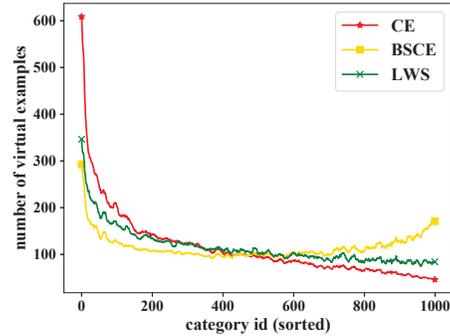

Figure 4. Smoothed virtual example distributions of different teachers on ImageNet-LT (a long-tailed version of ImageNet), with the temperature $\tau = 2$. The category id is sorted based on the number of examples in each category, from head to tail. Both BSCE [21] and LWS [15] generate flatter distributions than the baseline cross entropy method.
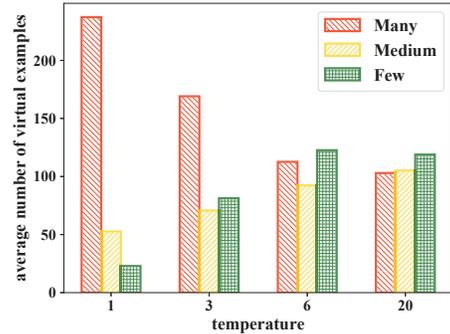

Figure 5. The virtual example distribution becomes flatter when the temperature τ increases, illustrated on CIFAR-100-LT with imbalance factor 100. The model was trained with BSCE [21]. Based on our rule-of-thumb, $\tau = 6$ is a proper temperature.

knowledge distillation, the teacher's virtual example distribution t is an explicit supervision signal, while we have various knobs to directly tune this distribution towards a flatter one. We also provide a rule-of-thumb for determining the level of flatness.

To level the virtual example distribution, the temperature is in fact already a built-in weapon in KD. Eqn. (5) clearly tells us that when the temperature τ increases, the teacher signal t^τ will be more and more balanced. As [33] mentioned, when $\tau \to \infty$, t^τ will become a uniform distribution. Fig. 5 illustrates this trend when τ increases.

A very large temperature (*e.g.*, the last temperature in Fig. 5 with $\tau = 20$) does not deem useful in knowledge distillation, because then the supervision signal t^τ will be roughly uniform and contains little information.

Thus, we resort to a classic trick (power normalization) to further adjust the virtual example distribution without in-

creasing τ to an unreasonable range. The power normalization [20] simply converts a nonnegative real number x to its power x^p. For simplicity, we always set $p = 0.5$ in our experiments. That is, to level the teacher's virtual example distribution, we perform the following transformations:

$$t_k^\tau \longleftarrow \sqrt{t_k^\tau}, \qquad \forall\, 1 \le k \le C, \qquad (13)$$

$$t_i^\tau \longleftarrow \frac{t_i^\tau}{\sum_k t_k^\tau} \qquad \forall\, 1 \le i \le C. \qquad (14)$$

It is easy to find out that applying the power normalization with p amounts to multiplying $\frac{1}{p}$ to the temperature τ for the teacher's supervision signal $\boldsymbol{t}^\tau$ (and $p = 0.5$ means doubling τ for the teacher). The temperature for student $\boldsymbol{s}^\tau$, however, remains unchanged.

Next, we introduce a rule-of-thumb to choose the temperature τ. We want the virtual example distribution to be more balanced. But we also want to keep it to be relatively small (*e.g.*, $\tau < 10$). Hence, after training a teacher model, we will compute the virtual example distribution of $\boldsymbol{t}^\tau$ on the entire training set for different τ between 1 and 10, and with or without power normalization. We prefer distributions that are flat, specifically, whose average number of examples per category in the tail part is slightly higher than that in the head part. For example, applying this rule-of-thumb, we will choose $\tau = 6$ in Fig. 5, then temperature for student $\boldsymbol{s}^\tau$ should be 3 if power normalization ($p = 0.5$) is used for the teacher $\boldsymbol{t}^\tau$. Please note that computing the virtual example distributions does *not* involve any fine-tuning or training of networks, hence it is very efficient—we need to simply transform the vectors $\boldsymbol{t}^\tau$ using different τ values and then normalize them.

One final thing to note is that in a long-tailed setting, using $\tilde{\boldsymbol{t}}$ is at best suboptimal, because the $\boldsymbol{y}$ in Eqn. (6) is distributed in a long-tailed fashion and we cannot utilize the adjusting capability of the temperature τ. Thus, in knowledge distillation we use the BSCE loss $L_{\mathrm{CE}}(\boldsymbol{y}, \boldsymbol{s}^{\mathrm{BSCE}})$. The overall DiVE loss function is then

$$\begin{aligned} &L_{\mathrm{DiVE}}(\boldsymbol{y}, \boldsymbol{s}^{\mathrm{BSCE}}) \\ &= (1 - \alpha) L_{\mathrm{CE}}(\boldsymbol{y}, \boldsymbol{s}^{\mathrm{BSCE}}) + \alpha \tau^2 L_{\mathrm{KL}}(\boldsymbol{t}^\tau, \boldsymbol{s}^\tau). \end{aligned} \quad (15)$$

The first term is the BSCE loss, in which the student's soft logits $\boldsymbol{s}^{\mathrm{BSCE}}$ does not involve the temperature. The second term is the KD term, in which $\boldsymbol{t}^\tau$ uses a temperature τ and possibly followed by a power normalization ($p = 0.5$), while $\boldsymbol{s}^\tau$ only uses the temperature τ but does not apply the power normalization.

With all necessary components ready, the simple DiVE pipeline is summarized in Algorithm 1.

Note that we choose BSCE as our teacher model because it has a good starting point for virtual example distribution (*c.f.* Fig. 4), and at the same time it is simple in implementation. But the teacher model can be trained by any other

---

Algorithm 1: The DiVE pipeline

Input: A long-tailed training set D.
1. Use BSCE to train a teacher model on D;
2. Use the rule-of-thumb to determine τ and determine whether to use the power normalization;
3. Transform teacher's soft logits as $\boldsymbol{t}^\tau$ accordingly;
4. Train a DiVE model by minimizing Eqn. (15).

---

method. Without special instructions, the teacher and the student use the same model architecture in our experiments.

4. Experimental Results

We now validate DiVE on various long-tailed datasets, with the empirical settings and implementation details in Sec. 4.1, evaluation settings in Sec. 4.2, and main results in Sec. 4.3. Sec. 4.4 analyzes various aspects of DiVE.

4.1. Datasets and implementations

We conduct experiments on three major benchmarks to evaluate the effectiveness of our proposed DiVE.

CIFAR-100-LT. These long-tailed versions of CIFAR-100 [3] follow an exponential decay in sample sizes across different classes with various imbalance factor β. we use $\beta = 10, 50, 100$ in our experiments. ResNet-32 [11] is used as the backbone network. We use the same training recipe as [34] with standard CIFAR data augmentation.

ImageNet-LT. They are long-tailed versions of ImageNet [17]. We use ResNeXt-50 [30] as the backbone in all experiments. For training strategies, we follow [15].

iNaturalist2018. iNaturalist2018 [4] is a large-scale real-world datasets with severe long-tail problems. We select ResNet-50 [11] as the backbone network and apply the similar training strategies with ImageNet-LT. 90 and 200 epochs results are reported.

For more details of the datasets and implementation, please refer to our supplementary materials.

4.2. Evaluation setups and comparison methods

After long-tailed training, we evaluate the models on the corresponding balanced validation/test dataset, and report the commonly used top-1 accuracy over all classes, denoted as "All". We also report the top-5 accuracy on iNaturalist2018 to evaluate the robustness of the methods. To better understand the methods' abilities on categories with different number of examples, we follow [15, 17] to split the categories into three subsets and report the average accuracy rates in these three subsets: *Many-shot* ($>$100 images), *Medium-shot* ($20 \sim 100$ images), and *Few-shot* ($<$20 images), which are also called the head, medium and tail categories, respectively.

We compare DiVE with two groups of methods:

Table 1. Top-1 accuracy (%) on CIFAR-100-LT. The "†" symbol denotes results copied directly from [34].

Methods	Imbalance factor		
	100	50	10
CE	38.35	42.41	56.51
Focal† [16]	38.41	44.32	55.78
BSCE	42.39	47.60	58.38
LFME [29]	43.80	-	-
LDAM-DRW [1]	42.04	46.62	58.71
BBN [34]	42.56	47.02	59.12
Meta-learning [14]	44.70	50.08	59.59
LDAM-DRW+SSP [31]	43.43	47.11	58.91
TDE [26]	44.10	50.30	59.60
DiVE	**45.35**	**51.13**	**62.00**

- **Baseline methods.** Networks trained with the standard cross-entropy loss and the focal loss [16] are used as baselines in this group. Also, the balanced softmax method proposed in [21] (which also trains all our teacher networks) is compared in all experiments.
- **State-of-the-art methods.** We also compare our DiVE method with recently proposed state-of-the-art methods, like De-confund-TDE [26] ("TDE" in short) and RIDE [27]. We further apply our DiVE to RIDE and get RIDE-DiVE.

4.3. Main results

We show our experimental results on the three datasets one by one, and finally RIDE-DiVE. For more experiments, please refer to our supplementary materials.

Results on CIFAR-100-LT. Table 1 shows the experimental results on long-tailed CIFAR-100 with various imbalance factors ranging from 10 to 100. The proposed DiVE method consistently achieves the best results under all imbalance factors, and it outperforms the state-of-the-art method De-confund-TDE [26] by a large margin.

Although knowledge distillation is also used in the LFME method [29], DiVE utilizes the abundant head-class samples to produce virtual examples for tail classes, thus enjoys the benefit of information from the entire dataset. Table 1 clearly shows that DiVE outperforms LFME by a large margin (1.55 percentage points).

Results on ImageNet-LT. We further evaluate DiVE on the ImageNet-LT dataset, with results in Table 2. We also report the average accuracy details of each category subsets.

DiVE almost obtains consistently higher accuracy rates than all compared methods in all comparisons (*Many*, *Medium*, *Few*, and "All").

DiVE also beats the teacher model BSCE in all three subsets, and its accuracy loss in the *Many* subset is less than 1%. On the contrary, the compared methods often lose accuracy in one of the subsets, and their accuracy loss in the *Many* subset is both consistent and significant.

Table 2. Comparison with state-of-the-art methods on ImageNet-LT. A "†" symbol denotes results copied from [15], and a "*" symbol denotes results obtained by running author-provided code.

Methods	Many	Medium	Few	All
CE	**65.02**	37.07	8.07	43.89
BSCE	60.92	47.97	29.79	50.48
OLTR† [17]	-	-	-	46.30
τ-norm [15]	59.10	46.90	30.70	49.40
LWS [15]	60.20	47.20	30.30	49.90
TDE [26]	62.70	48.80	**31.60**	51.80
TDE*	62.56	47.83	29.91	51.06
DiVE	64.06	**50.41**	31.46	**53.10**

Table 3. Results on the large-scale long-tailed iNaturalist2018 dataset. We present results when trained for 90 & 200 epochs, except for BBN [34] (which were trained 90 & 180 epochs). BBN's top-5 accuracy is from the author-released checkpoint. A "†" symbol denotes results copied directly from [1].

Methods	90 epochs		200 epochs	
	top-1	top-5	top-1	top-5
CE	62.60	83.44	-	-
CB-Focal† [3]	61.12	81.03	-	-
BSCE	65.35	83.36	67.84	85.45
LDAM-DRW† [1]	68.00	85.18	-	-
BBN [34]	66.29	85.57	69.65	87.64
Meta-learning [14]	67.55	86.17	-	-
LWS [15]	65.90	-	69.50	-
cRT+SSP [31]	68.10	-	-	-
DiVE	**69.13**	**86.85**	**71.71**	**88.39**

Results on iNaturalist2018. To verify the performance of DiVE in real world long-tailed circumstances, we conduct experiments on the iNaturalist2018 dataset. Table 3 shows the overall accuracy results computed using all categories. Following [15], besides 90 epochs, we train for more epochs (200 epochs) to get further improvement. We gain 2.58% and 1.54% on top-1/top-5 accuracy from that. In terms of top-1 accuracy, DiVE is at least 1 percentage point higher than all compared methods in both settings.

We further break the accuracy statistics into three groups, and the results are in Table 4. BBN hurts the *Many-shot* subset a lot to enhance the *Medium-shot* and *Few-shot*, while LWS has very similar accuracy with the BSCE baseline in all subsets. On the contrary, DiVE outperforms BSCE consistently and significantly in all three subsets. DiVE's accuracy drop in the *Many* subset from the baseline CE method is also much smaller than other methods.

Results of RIDE-DiVE. Our DiVE can be easily deployed on any existing method. Following [27], we first use RIDE with 6 experts to generate the virtual examples, then distill to a 4 experts student model. Results are in Table 5. It is worth noting that our teacher networks (using BSCE) is much inferior to *the RIDE teacher* in two datasets, but our student networks surpass both the teacher and RIDE, setting

Table 4. Accuracy (%) on the three subsets of iNaturalist2018. The 90-epoch model was used in this table. BBN's results are from the author-released checkpoint.

Methods	Many	Medium	Few	All
CE	**73.08**	63.74	58.41	62.60
BSCE	65.20	65.38	65.38	65.35
BBN [34]	49.49	**70.87**	65.31	66.43
LWS [15]	65.00	66.30	65.50	65.90
DiVE	70.63	70.01	**67.58**	**69.13**

Table 5. Accuracy (%) compared with RIDE [27] on three datasets. ResNet-32, ResNeXt-50 and ResNet-50 are used as backbones, respectively. "C100-LT" is short for CIFAR-100-LT ($\beta = 100$), "IN-LT" for ImageNet-LT, and "iNat18" for iNaturalist2018. Arrows indicate whether student's accuracy is higher or lower than the teacher network. iNat18 is trained for 100 epochs.

Methods	C100-LT	IN-LT	iNat18
RIDE teacher	50.20	57.50	72.80
RIDE [27]	49.10 ↓	56.80 ↓	72.60 ↓
RIDE-DiVE teacher	51.07	55.60	68.79
RIDE-DiVE	**51.66** ↑	**57.12** ↑	**73.44** ↑

Table 6. Effects of balancing the virtual example distribution.

	BSCE	$\tilde{t}/t^{\tau}$	τ	power	100	50	10
CE	-	-	-	-	38.35	42.41	56.51
# 1	✓	t^{τ}	3	✓	**45.35**	**51.13**	**62.00**
# 2	✗	$\tilde{t}$	1	✓	44.55	49.69	61.62
# 3	✗	t^{τ}	3	✓	44.50	50.20	61.28
# 4	✗	t^{τ}	1	✗	43.25	47.64	60.07
# 5	✗	$\tilde{t}$	1	✗	41.59	47.10	59.10
# 6	✗	$\tilde{t}$	3	✓	43.22	48.51	60.59

new state-of-arts for these datasets.

4.4. Further Analyses

In this final part, we analyze various components in our DiVE method, with results mainly presented in Table 6 and Fig. 6, all from experiments on CIFAR-100-LT ($\beta = 100$).

The first row "CE" is the baseline that only uses a normal cross entropy loss, and row #1 to #6 study a few variants of the DiVE method. The column "BSCE" means whether using the first loss term in Eqn. (15) or not. $\tilde{t}$ means using Eqn. (6), while t^{τ} means not to factor in groundtruth labels. τ is the temperature value, and "power" means whether the power normalization is used or not. Hence, row #1 is DiVE with the temperature $\tau = 3$ and power normalization.

Firstly, as Sec. 3.1 shows, KD ≈ DLDL. In row #3, we remove the first term in the right hand side of Eqn. (15) and only use the DLDL loss. Without the BSCE loss term, DiVE's accuracy drops (but less than 1% in all three cases). However, cross compare row #3 with results in Table 1, the DLDL-only version of DiVE is still better than most compared methods. This fact shows that *distilling the virtual examples alone* are very effective in long-tailed recognition.
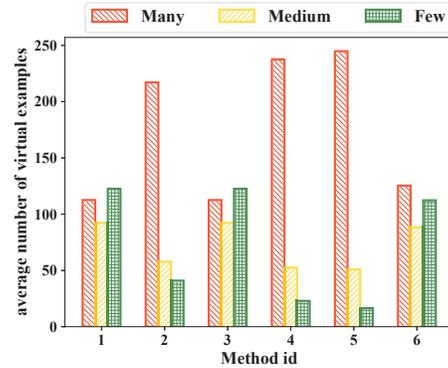

Figure 6. Virtual example distribution of different method id.

A close examination of Fig. 6 shows that row #1 (DiVE) indeed leads to a balanced virtual example distribution, and the tail is slightly higher than the heads, which matches our rule-of-thumb.

In row #5, we factor the groundtruth label into the teacher's supervision signal, as directed by Eqn. (6) ($\alpha = 0.5$ and $\tau = 1$). The BSCE loss is removed, too. Its results are the worst in all DiVE variants, and in Fig. 6 its distribution is the most imbalanced.

On top of row #5, we can add the power normalization (row #2) and a further temperature $\tau = 3$ (row #6). As Fig. 6 shows, both operations make the virtual examples distribute more evenly, at the same time their accuracy rates are also improved. Row #2 results are similar to DiVE without BSCE (row #3), which corroborates with the derivation in Sec. 3.1. The fact that all rows using $\tilde{t}$ are worse than row #1 (DiVE, which uses t^{τ}) supports our decision of not using $\tilde{t}$ (*c.f.* Sec. 3.4).

Besides, the fact that row #4 has the lowest accuracy among variants that use t^{τ} once again shows that a balanced virtual example distribution is important.

5. Conclusions

In this paper, we proposed the DiVE method to distill the predictions of a teacher model (as the virtual examples) into a student model, which leverages interactions among different categories for long-tailed visual recognition: virtual examples from head categories will help recognize tail categories, even if these categories are not correlated. Therefore, employing knowledge distillation upon virtual examples was able to alleviate the extreme imbalance for long-tailed data, particularly for the tail classes. Also, in order to further improve the distillation accuracy, we provided a rule-of-thumb for adjusting the distribution of the virtual examples towards flat. DiVE has achieved the best results on long-tailed benchmarks, including the large-scale iNaturalist. In the future, we attempt to extend our DiVE into handling the long-tailed detection problem.

References

[1] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. In *Adv. Neural Inform. Process. Syst. 32*, 2019. 1, 2, 7

[2] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002. 1, 2

[3] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 9268–9277, 2019. 1, 2, 6, 7

[4] Yin Cui, Yang Song, Chen Sun, Andrew Howard, and Serge Belongie. Large scale fine-grained categorization and domain-specific transfer learning. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 4109–4118, 2018. 1, 6

[5] Chris Drummond and Robert C Holte. C4.5, class imbalance, and cost sensitivity: Why under-sampling beats over-sampling. In *Workshop on learning from imbalanced datasets II*, volume 11, pages 1–8, 2003. 1, 2

[6] Bin-Bin Gao, Chao Xing, Chen-Wei Xie, Jianxin Wu, and Xin Geng. Deep label distribution learning with label ambiguity. *IEEE Trans. Image Process.*, 26(6):2825–2838, 2017. 3, 4

[7] Xin Geng. Label distribution learning. *IEEE Transactions on Knowledge and Data Engineering*, 28(7):1734–1748, 2016. 4

[8] Agrim Gupta, Piotr Dollar, and Ross Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5356–5364, 2019. 1

[9] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In *International Conf. Intelligent Computing*, pages 878–887, 2005. 1, 2

[10] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009. 1, 2

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 770–778, 2016. 6

[12] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 1, 3

[13] Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. Learning deep representation for imbalanced classification. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5375–5384, 2016. 2

[14] Muhammad Abdullah Jamal, Matthew Brown, Ming-Hsuan Yang, Liqiang Wang, and Boqing Gong. Rethinking class-balanced methods for long-tailed visual recognition from a domain adaptation perspective. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 7610–7619, 2020. 1, 7

[15] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. In *Int. Conf. Learn. Represent.*, 2020. 1, 2, 5, 6, 7, 8

[16] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Int. Conf. Comput. Vis.*, pages 2999–3007, 2017. 1, 2, 7

[17] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X. Yu. Large-scale long-tailed recognition in an open world. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 2, 6, 7

[18] Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. *arXiv preprint arXiv:2007.07314*, 2020. 2

[19] Ravi Teja Mullapudi, Fait Poms, William R Mark, Deva Ramanan, and Kayvon Fatahalian. Background splitting: Finding rare classes in a sea of background. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 8043–8052, 2021. 3

[20] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the Fisher Kernel for Large-Scale Image Classification. In *Eur. Conf. Comput. Vis.*, LNCS 6314, pages 143–156, 2010. 6

[21] Jiawei Ren, Cunjun Yu, Xiao Ma, Haiyu Zhao, Shuai Yi, et al. Balanced meta-softmax for long-tailed visual recognition. In *Adv. Neural Inform. Process. Syst. 33*, pages 4175–4186, 2020. 2, 5, 7

[22] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet large scale visual recognition challenge. *Int. J. Comput. Vis.*, 115(3):211–252, 2015. 1

[23] Li Shen, Zhouchen Lin, and Qingming Huang. Relay back-propagation for effective learning of deep convolutional neural networks. In *Eur. Conf. Comput. Vis.*, LNCS 9911, pages 467–482, 2016. 1, 2

[24] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2818–2826, 2016. 4

[25] Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. Equalization loss for long-tailed object recognition. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 11662–11671, 2020. 1, 2

[26] Kaihua Tang, Jianqiang Huang, and Hanwang Zhang. Long-tailed classification by keeping the good and removing the bad momentum causal effect. In *Adv. Neural Inform. Process. Syst. 33*, pages 1513–1524, 2020. 2, 7

[27] Xudong Wang, Long Lian, Zhongqi Miao, Ziwei Liu, and Stella X Yu. Long-tailed recognition by routing diverse distribution-aware experts. In *Int. Conf. Learn. Represent.*, 2021. 2, 7, 8

[28] Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. Learning to model the tail. In *Adv. Neural Inform. Process. Syst. 30*, pages 7029–7039, 2017. 2

[29] Liuyu Xiang, Guiguang Ding, and Jungong Han. Learning from multiple experts: Self-paced knowledge distillation for long-tailed classification. In *Eur. Conf. Comput. Vis.*, pages 247–263, 2020. 2, 3, 7

[30] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1492–1500, 2017. 6

[31] Yuzhe Yang and Zhi Xu. Rethinking the value of labels for improving class-imbalanced learning. In *Adv. Neural Inform. Process. Syst. 33*, pages 19290–19301, 2020. 2, 3, 7

[32] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 4133–4141, 2017. 3

[33] Li Yuan, Francis Eh Tay, Guilin Li, Tao Wang, and Jiashi Feng. Revisiting knowledge distillation via label smoothing regularization. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3903–3911, 2020. 3, 4, 5

[34] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. BBN: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 9719–9728, 2020. 1, 2, 6, 7, 8

[35] Linchao Zhu and Yi Yang. Inflated episodic memory with region self-attention for long-tailed visual recognition. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 4344–4353, 2020. 2