Efficient One Pass Self-distillation with Zipf's Label Smoothing

Jiajun Liang[†], Linze Li, Zhaodong Bing, Borui Zhao, Yao Tang, Bo Lin, and Haoqiang Fan

MEGVII Technology
{liangjiajun,lilinze,bingzhaodong,zhaoborui,tangyao02,linbo,fhq}@megvii.com

Abstract. Self-distillation exploits non-uniform soft supervision from itself during training and improves performance without any runtime cost. However, the overhead during training is often overlooked, and yet reducing time and memory overhead during training is increasingly important in the giant models' era. This paper proposes an efficient self-distillation method named Zipf's Label Smoothing (Zipf's LS), which uses the on-the-fly prediction of a network to generate soft supervision that conforms to Zipf distribution without using any contrastive samples or auxiliary parameters. Our idea comes from an empirical observation that when the network is duly trained the output values of a network's final softmax layer, after sorting by the magnitude and averaged across samples, should follow a distribution reminiscent to Zipf's Law in the word frequency statistics of natural languages. By enforcing this property on the sample level and throughout the whole training period, we find that the prediction accuracy can be greatly improved. Using ResNet50 on the INAT21 fine-grained classification dataset, our technique achieves +3.61% accuracy gain compared to the vanilla baseline, and 0.88% more gain against the previous label smoothing or self-distillation strategies. The implementation is publicly available at `https://github.com/megvii-research/zipfls`.

Keywords: Knowledge Distillation, Self Distillation, Label Smoothing, Image Classification, Zipf's Law

1 Introduction

A major trend in the study of multi-class classification models is to replace the one-hot encoding label with more informative supervision signal. This line of thought witnessed proliferation of great training techniques to improve the accuracy of a network without any runtime cost, the Knowledge Distillation being the most famous of them. Perhaps the most counter-intuitive discovery in this direction is the effectiveness of Self Distillation methods [7,27–29,32] where a network even benefits from predictions of its own.

Self distillation simplifies the two-stage knowledge distillation framework by distilling knowledge from itself instead of from the pretrained teacher, and still
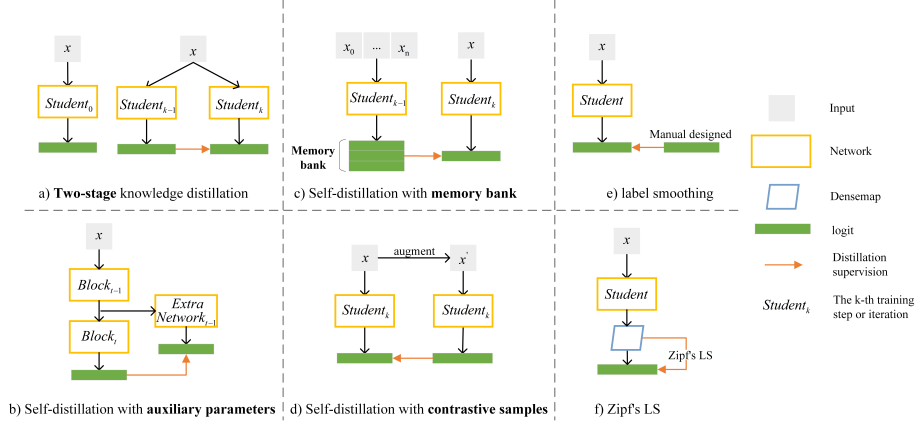
Fig. 1: Comparison between different soft label generation methods. a) **Two-stage** knowledge distillation method [7, 12]. b) **Auxiliary parameters** [8, 13, 33]. c) Progressive distillation with **memory bank** storing past predictions of the entire dataset [2, 14, 31]. d) Contrastive samples [27, 29] with **twice training iterations**. e) Label smoothing methods [19, 28], with **uninformative** manual designed distributions. f) **Efficient one-pass** Zipf's Label Smoothing method, generating sample-level non-uniform soft labels almost without additional cost during training

improves performance significantly without extra cost in inference time. However, the overhead during training in self-distillation is often overlooked, and yet reducing time and memory overhead in training is increasingly important in today's giant model era. Fig. 1 shows several knowledge distillation paradigms, and self distillation methods rely on additional contrastive training instances, auxiliary parameters or intermediate dumped results for each training sample, which could double the training time and bring non-negligible memory overhead.

This paper aims to find efficient techniques which generate non-uniform supervision signals as informative as expensive self distillation approaches. The starting point of our construction is the observation of the network's soft-max output values. The class that corresponds to the final categorical output usually has the highest value, but the scores for other classes (which we call the non-target classes), containing important information of the network's understanding of the input image, also play an important role. Indeed, we surmise that **a good-performing network should obey certain laws in the non-zero prediction values they make on the non-target classes**.

We postulate that a significant part of the efficacy of the distillation techniques comes from enforcing the prediction scores into a shape that best balances between being "sharp" (so that the final prediction is unambiguous) and being "soft" (so that inter-class correlation are respected). We test this hypothesis by inventing a technique, which only uses the on-the-fly prediction of a network to generate a soft supervision label that conforms to our designated distribution
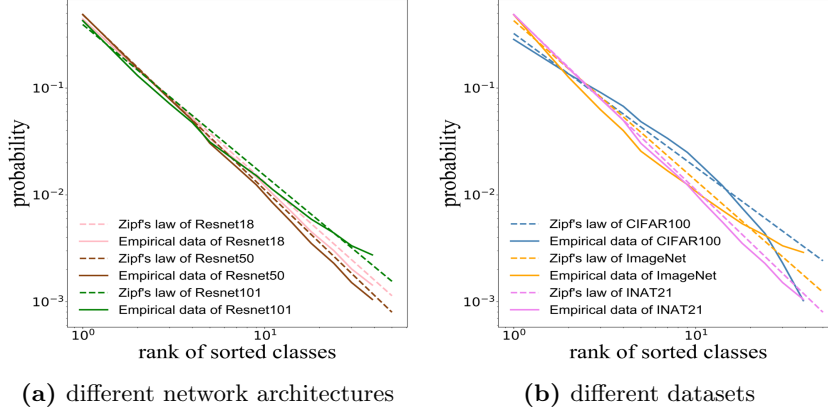
(a) different network architectures   **(b)** different datasets

Fig. 2: Distribution of sorted softmax scores from duly-trained networks of a)different architectures on INAT-21 and b) ResNet-50 on different datasets. The average distribution of sorted softmax values (the solid lines) well follows Zipf's law (the dashed lines of the same color). The probability-rank relations form straight lines in the log-log plot. This inspires us to design the soft supervision similar to this shape

law, and showing that this simple strategy already harvests or even surpasses the performance gains of many other more complex distillation techniques.

Specifically, we rank the output classes according to the feature output of the multi-class classification network, and assign to each class a target value according to Zipf's Law Distribution:

$$\boldsymbol{p} \propto \boldsymbol{r}^{-\alpha}$$

where $\boldsymbol{p}$ is the confidence distribution of different classes, $\boldsymbol{r}$ is our sorted rank index of classes (integer values starting from 1), and α is a hyper-parameter that controls the rate of decay. Divergence to this smoothed label is added as a loss term in complement to the usual cross-entropy with the one-hot encoded hard label.

The choice of Zipf's Law is not arbitrary. Indeed, we experimentally find out that when a network is trained to its convergence state, the average distribution of sorted softmax values well follows this law (see Fig. 2). By explicitly enforcing this shape of the distribution and giving supervision from the very beginning of training, the performance of the network boots by a significant amount.

Thus we propose an efficient and general plug-and-play technique for self distillation, named Zipf's Label Smoothing (Zipf's LS). Compared to other techniques (see Table 1), our method enjoys the advantage of incurring almost zero additional cost (during inference or training) as label smoothing and at the same time strongly preserves the performance gains of the self-distillation. To summarize, our contributions are as follows:

Table 1: Comparison between different soft label generation methods. Our Zipf's Label Smoothing generates sample-level non-uniform soft labels with little cost during training. For simplicity, this table only shows top1-accuracy(Acc) from ResNet-18 on TinyImageNet(Tiny) and ResNet-50 on ImageNet(IMT). More comprehensive results on other models and datasets are reported in the remaining sections of this paper. Memory cost and training time test experiments are conducted on ImageNet using 4 2080Ti GPUs with batch size 16

Method	Non-Uniform	w/o Pretrain Teacher	w/o Contrastive Samples	w/o Auxiliary Parameters	CPU Memory Cost	GPU Memory Cost	Training time per epoch	Tiny Acc (gain)	IMT Acc (gain)
Baseline	✗	✓	✓	✓	18G	6.8G	1.82h	56.41	76.48
BAN [7]	✓	✗	✓	✗	18.9G	7.4G	2.67h	(+2.24)	(+0.05)
BYOT [33]	✓	✓	✓	✗	18.2G	38.5G	10.36h	(+1.43)	(+0.51)
PS-KD [14]	✓	✓	✗	✓	27.4G	9.8G	2.18h	(+1.81)	(+0.18)
DDGSD [27]	✓	✓	✗	✓	18.2G	7.3G	4.30h	(+2.11)	(+0.46)
CS-KD [29]	✓	✓	✗	✓	18G	6.8G	2.86h	(+1.97)	(+0.30)
LS [24]	✗	✓	✓	✓	18G	6.8G	1.83h	(+0.48)	(+0.19)
TF-KD [28]	✗	✓	✓	✓	18G	6.8G	1.82h	(+0.26)	(+0.08)
Ours	✓	✓	✓	✓	18G	6.8G	1.83h	**(+2.84)**	**(+0.77)**

- We find that distribution of non-target soft-max values of duly trained model fits well to Zipf's law, which could be used as a regularization criterion in the entire self-training process.
- We propose Zipf's Label Smoothing method, an efficient self distillation training technique without relying on additional contrastive training instances or auxiliary parameters.
- We verify our method on comprehensive combinations of models and datasets (including popular ResNet and DenseNet models, CIFAR, ImageNet and INAT classification tasks) and show strong results.

2   Related Work

2.1   Label Smoothing

The one-hot label is sub-optimal because objects from more than one class occur in the same image. Label Smoothing [24] (LS) is proposed to smooth the hard label to prevent over-confident prediction and improve classification performance. Müller et al. [19] found that intra-class distance in feature space is more compact when LS is used, which improves generalization. To obtain non-uniform soft labels, Zhang et al. [31] proposed the online label smoothing method (OLS) by maintaining the historical predictions to obtain the class-wise soft label. Yuan et al. [28] discussed the relationship between LS and knowledge distillation, and proposed a teacher-free knowledge distillation (Tf-KD) method to get better performance than LS. Label smoothing has become one of the best practices in the current deep learning community [11], but the paradigm of using uniform distribution for the non-target classes limits further improvement of performance.

2.2   Knowledge Distillation

Instead of imposing a fixed prior distribution, knowledge distillation was first proposed by Hinton in [12] to provide sample-level non-uniform soft labels. They demonstrated that the "dark knowledge" lies in the output distributions from a large capacity teacher network and benefits the student's representation learning. Recent works mainly explored to better transfer the "dark knowledge" and improve the efficiency from various aspects, such as reducing the difference between the teacher and student [3, 5, 18, 34], designing student-friendly architecture [16, 20], improving the distillation efficiency [7, 14, 27, 29] and explaining the distillation's working mechanism [1, 23].

In this work, we focus on how to transfer the "dark knowledge" in an almost free manner. Furlanello et al. [7] proposed to improve the performance of the student network by distilling a teacher network with the same architecture. However, it is still a two-stage approach, which first trains the teacher and then distills knowledge to the student. To reduce the training time, many self-distillation methods were proposed. They gain soft label supervision on the fly without the pretraining step.

2.3   Self Distillation

There are two categories of self-distillation techniques, namely the auxiliary parameter methods [2, 8, 13, 30, 33] and contrastive sample methods [14, 27, 29, 31]. Auxiliary model methods exploit additional branches to get extra predictions besides the main-branch prediction for soft label supervision at the cost of more parameters overhead. For example, Knowledge Distillation via Collaborative Learning (KDCL) [8] trained multiple parallel student networks at the same time and ensemble the output as extra soft label supervision for each parallel student network. On the other hand, contrastive sample methods get soft label supervision at the cost of additional data augmentation, enlarged batch size, or complex sampling strategy. The examples are Data-distortion Guided Self-Distillation (DDGSD) [27] which gains soft labels from different augmented views from the same instance and Regularizing Class-wise Predictions via Self-knowledge Distillation (CS-KD) [29] which gathers data from other samples of the same class.

As summarized above, Label Smoothing and Knowledge Distillation are two major techniques to acquire informative soft labels. However, the Label Smoothing methods are limited by the uniform hypothesis, while the Knowledge Distillation methods require much more memory or computation overhead. Our work aims to improve upon these issues.

3   Method

Zipf's Label Smoothing aims to combine the best of the two worlds, the efficiency of teacher-free label smoothing and the informative soft label from self
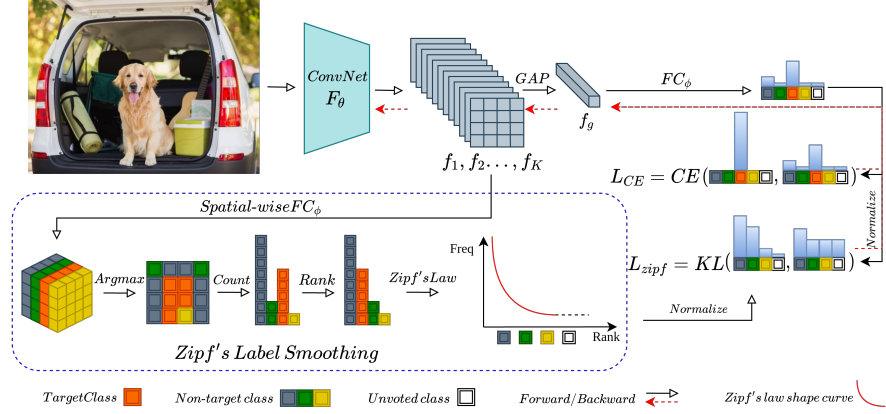
Fig. 3: The overall Zipf's soft label generation and training framework. The blue dashed box outlines the soft label generation process. We apply a shared-classifier on dense feature maps and count the number of argmax values from the dense prediction, which could provide ranking information to Zipf's law distribution generation. For the non-occurred classes with zero count, we give uniform constant energy to them. L_{zipf} is Kullback–Leibler divergence between the prediction of non-target classes and zipf's soft label, combined with L_{CE} of the hard label to provide gradient to representation learning

distillation. It generates non-uniform supervision signals from on-the-fly prediction of the network as shown in Fig. 3. Our method is inspired by the observation that the value and rank of the softmax output from the duly trained network follow a distribution reminiscent to the Zipf's Law on average as shown in Fig. 2, which could be applied as a shape prior to the softmax prediction during the whole training period. To apply Zipf's Law to soft-label generation, ranking information of output categories is needed. We propose Dense Classification Ranking which utilizes local classification results to rank the categories. Finally, KL-divergence between prediction and Zipf's soft label within non-target classes is measured to provide more informative gradients for representation learning.

3.1   Zipf's Law

Zipf's law is an empirical law which states that the normalized frequency of an element should be inversely proportional to the rank of the element, firstly discovered by G.Zipf on linguistic materials [21]. It can be described by an equation

as:

$$f(r) = \frac{r^{-\alpha}}{\sum_{r=1}^{N} r^{-\alpha}}$$

$$log f(r) = -\alpha \log r - \log(\sum_{r=1}^{N} r^{-\alpha})$$

(1)

where r is the rank of the element, N is the total number of elements, f is the frequency and α is a constant larger than zero which controls the decay rate.

An interesting discovery is that the outputs of softmax networks follow Zipf's Law [21] when the network is trained to its convergence state, and this pattern consistently emerges in different datasets and models as shown in Fig. 2. We could exploit this shape prior to design a simple and efficient self-distillation algorithm. To generate a soft label using Zipf's distribution, ranking information of categories is a must which is almost impossible to get from annotation.

3.2   Dense Classification Ranking

To realize the Zipf distribution we need to find a way to properly rank the output categories. The naive thought is to directly sort the softmax prediction of a sample, which we call the logit-based ranking method. Though this method already generates performance gain (as shown in section 4.4), we find a finer treatment of the relative ranking of the top classes to be beneficial.

Common image classification convolution networks extract a feature map $\boldsymbol{F}$ from an image sample. A global average pooling (GAP) is then applied to $\boldsymbol{F}$. The fully-connected layer FC and soft-max operation will output the logits $\boldsymbol{z}$ and the final prediction $\boldsymbol{p}$.

If we used FC directly on every pixel of the dense feature map $\boldsymbol{F}$, we cat get local classification results $\boldsymbol{p}^L$:

$$\boldsymbol{p}_k^L = Softmax(FC_\phi(\boldsymbol{F}_k)), k = 1, 2, ..., H \times W \qquad (2)$$

where L is just a mark to show the difference between global prediction $\boldsymbol{p}$ and local predictions $\boldsymbol{p}^L$. These predictions give a more complete description of what the image contains, as the object of the target class usually only occupies part of the image. We take this information into account by identifying the individual top-1 class from each location as a vote and aggregate the votes into a histogram. The classes are finally ranked by their frequency of appearance in the histogram. The remaining classes will share the same lowest rank.

3.3   Zipf's Loss

Zipf's loss L_{Zipf} of image sample $\boldsymbol{x}$ and label y is defined as KL-divergence D_{KL} between normalized non-target prediction $\hat{\boldsymbol{p}}$ and the synthetic Zipf's soft label $\tilde{\boldsymbol{p}}$ generated from dense classification ranking:

$$\hat{\boldsymbol{p}}_c = \frac{\exp\left(\boldsymbol{z}_c\right)}{\sum_{m=1,m\neq y}^{C} \exp\left(\boldsymbol{z}_m\right)}$$

$$L_{Zipf}(\boldsymbol{x},y) = D_{KL}(\tilde{\boldsymbol{p}}||\hat{\boldsymbol{p}}) = \sum_{c=1,c\neq y}^{C} \tilde{\boldsymbol{p}}_c log\frac{\tilde{\boldsymbol{p}}_c}{\hat{\boldsymbol{p}}_c} \tag{3}$$

The synthetic Zipf's label $\tilde{\boldsymbol{p}}$ of non-target class c should follow the equation 4 of Zipf's law with the corresponding rank $\boldsymbol{r}_c$:

$$\tilde{\boldsymbol{p}}_c = \frac{\boldsymbol{r}_c^{-\alpha}}{\sum_{m=1,m\neq y}^{C} \boldsymbol{r}_m^{-\alpha}} \tag{4}$$

where α is a hyper-parameter that controls the shape of the distribution.

The L_{CE} is the standard cross entropy loss with one-hot ground-truth label. The combined loss function is as follow:

$$Loss(\boldsymbol{x},y) = L_{CE}(\boldsymbol{x},y) + \lambda L_{Zipf}(\boldsymbol{x},y) \tag{5}$$

L_{CE} encourages the prediction to be sharp and confident in the target class, while L_{Zipf} regularized the prediction to be soft within non-target classes. λ is a hyper-parameter which control the regularization strength.

Comparison with Uniform Label Smoothing. The gradient with respect to non-target logits for L_{zipf} and L_{LS} are shown as:

$$\frac{\partial L_{Zipf}(\boldsymbol{x},y)}{\partial \boldsymbol{z}_c} = \begin{cases} 0 & c = y \\ \hat{\boldsymbol{p}}_c - \tilde{\boldsymbol{p}}_c & c \neq y \end{cases}$$

$$\frac{\partial L_{LS}(\boldsymbol{x},y)}{\partial \boldsymbol{z}_c} = \begin{cases} \boldsymbol{p}_c - (1-\beta) & c = y \\ \boldsymbol{p}_c - \frac{\beta}{C-1} & c \neq y \end{cases} \tag{6}$$

Label smoothing generates a soft label with a uniform value $\frac{\beta}{C-1}$ for all non-target classes and $1 - \beta$ for the target class, where C is the total number of classes. However, label smoothing suppresses predictions of high-ranked classes or promotes predictions of low-ranked classes to the same level since β is constant and rank-irrelevant, which is conceptually sub-optimal.

Our Zipf's loss is rank-relevant compared with label smoothing. In non-target classes, it encourages the high-ranked classes to keep larger predictions than low-ranked ones. Zipf's law distribution shows empirical success in our experiments against other rank-relevant distributions such as linear decay, more details are shown in Section 4.4.

4   Experiment

Table 2: Top-1 accuracy (%) on CIFAR100, TinyImageNet image classification tasks with various model architectures. We report the mean and standard deviation over five runs with different random seeds. Vanilla indicates baseline results from the standard cross-entropy, the best results are indicated in bold, and the second-best results are indicated by underline. The performances of state-of-the-art methods are reported for comparison

Method	CIFAR100	TinyImageNet	CIFAR100	TinyImageNet
	DenseNet121		ResNet18	
Vanilla	77.86±0.26	60.31±0.36	75.51±0.28	56.41±0.20
BAN [7]	78.39±0.14	59.34±0.60	76.96±0.04	58.65±0.83
BYOT [33]	78.93±0.05	60.54±0.02	77.15±0.03	57.84±0.15
PS-KD [14]	78.82±0.10	61.64±0.12	76.74±0.06	58.22±0.17
DDGSD [27]	78.18±0.02	60.80±0.30	76.48±0.13	58.52±0.12
CS-KD [29]	78.31±0.49	62.04±0.09	**78.01±0.13**	58.38±0.38
LS [19]	78.12±0.45	61.25±0.18	77.31±0.28	56.89±0.16
TF-KD [28]	77.68±0.21	60.17±0.57	77.29±0.15	56.67±0.05
Zipf's LS	**79.03±0.32**	**62.64±0.30**	77.38±0.32	**59.25±0.20**

4.1   Experimental Detail

Datasets. We conduct experiments in various image classification tasks to demonstrate our method's effectiveness and universality. Specifically, we use CI-FAR100 [15] and TinyImageNet[1] for small-scale classification tasks, and ImageNet [6] for large-scale classification task. We also verify fine-grained classification performance with INAT21 [25] using the "mini" training dataset.

Training setups. We followed the setups in recent related works [4,8,28,29] and the popular open-source work[2]. All experiments use MSRA initialization [10], SGD optimizer with 0.9 momentum, 0.1 initial learning rate, 1e-4 weight decay, and standard augmentations including random cropping, and flipping. For small-scale CIFAR100 and TinyImageNet datasets, we use 32x32 resized input images, 128 batch size, and step learning rate policy which decreased to 1/10 of its previous value at 100th and 150th in the overall 200 epochs. All small-scale experiments are trained with single GPU. For large-scale ImageNet and INAT21 datasets, we use 224x224 resized input images, 256 batch size, and step learning rate policy which decreased to 1/10 of its previous value at 30th, 60th, and 90th in the overall 100 epochs. All large-scale experiments are trained with 4 GPUs.

Hyper parameters. Our method has two hyperparameters in general, λ and α. λ controls regularization strength and α controls the decay shape of Zipf's

---

[1] https://www.kaggle.com/c/tiny-imagenet
[2] https://github.com/facebookresearch/pycls

distribution, which is set to 1.0 in all experiments. β is only recommended for small resolution datasets such as CIFAR100 and TinyImageNet, to exploit the higher resolution intermediate feature map and make a more reliable ranking. A detailed hyperparameters ablation study is shown in the supplement.

Table 3: Top-1 accuracy (%) comparison with state-of-the-art works. Experiments conduct on ImageNet, INAT21 image classification tasks with ResNet50, and on CIFAR100, TinyImageNet with DenseNet121.

Method	CIFAR100	TinyImageNet	ImageNet	INAT21
Vanilla	77.86	60.31	76.48	62.43
CS-KD [29]	<u>78.31</u>	<u>62.04</u>	<u>76.78</u>	<u>65.45</u>
LS [19]	78.12	61.25	76.67	65.16
TF-KD [28]	77.68	60.17	76.56	62.61
Zipf's LS	**79.03**	**62.64**	**77.25**	**66.04**

4.2   General Image Classification Tasks

First, we conduct experiments on CIFAR100 dataset and TinyImageNet dataset to compare with other related state-of-the-art methods, including self-knowledge distillation methods (BAN [7]), online knowledge distillation methods (DDGSD [27], CS-KD [29]), and label smoothing regularization method (label smoothing [19], TF-KD [28]). Table 2 shows the classification results of each method based on different network architectures. All experiments of the above methods keep the same setups for a fair comparison, details can be seen in 4.1 **Training setups**. For other hyper-parameters, we keep their original settings.

Comparison with two-stage knowledge distillation. Two-stage knowledge distillation methods improve model accuracy using their previous models' dark knowledge. These methods rely on a pretrained model, which means they take twice or more training time than our method. We make a big advantage beyond BAN [7] (one step) as Table 2 shows, specifically, Zipf's LS surpasses BAN [7] by 0.64% and 3.28% respectively on CIFAR100 and TinyImageNet based on DenseNet121.
Comparison with self-distillation. As one type self-distillation utilizing contrastive samples, DDGSD and CS-KD realize respectively exploiting instance-wise and class-wise consistency regularization techniques. The data process of DDGSD or the pair sample strategy of CS-KD brings double iterations per training epoch. As shown in Table 2, Zipf's LS achieves 0.9% and 0.73% gains compared with DDGSD [27] respectively on CIFAR100 and TinyImageNet based on ResNet18 without more training iterations. As another type self-distillation with auxiliary parameters, BYOT [33] squeeze deeper knowledge into lower network. Zipf's LS surpasses BYOT obviously on TinyImageNet.

Comparison with label smoothing regularization. Label smoothing [19] is a general effective regularizing method with manually designed soft targets. In Table 2, Zipf's LS beats label smoothing by 2.36% and 1.39% advantage on TinyImageNet respectively based on ResNet18 and DenseNet121.

Table 4: Top-1 accuracy (gain) (%) on ImageNet and INAT21 image classification tasks with various model architectures. Vanilla indicates baseline results from the cross-entropy, and the best results are indicated in bold

Architecture	Method	ImageNet	INAT21
ResNet18	Vanilla	70.47	54.31
	Label Smooth	70.53(+0.06)	55.17(+0.86)
	Zipf's LS	**70.73 (+0.26)**	**56.36 (+2.03)**
ResNet50	Vanilla	76.48	62.43
	Label Smooth	76.67(+0.19)	65.16(+2.73)
	Zipf's LS	**77.25 (+0.77)**	**66.04 (+3.61)**
ResNet101	Vanilla	77.83	65.60
	Label Smooth	78.12(+0.29)	67.14(+1.54)
	Zipf's LS	**78.58 (+0.75)**	**68.45 (+2.85)**
ResNeXt50_32x4d	Vanilla	77.54	66.36
	Label Smooth	77.72(+0.18)	67.51(+1.15)
	Zipf's LS	**78.07 (+0.53)**	**69.24 (+2.88)**
ResNeXt101_32x8d	Vanilla	79.51	70.35
	Label Smooth	79.69(+0.18)	71.52(+1.17)
	Zipf's LS	**80.03 (+0.52)**	**72.18 (+1.83)**
DenseNet121	Vanilla	75.56	63.75
	Label Smooth	75.59(+0.03)	64.60(+0.85)
	Zipf's LS	**75.98 (+0.42)**	**66.60 (+2.85)**
MobileNetV2	Vanilla	65.52	55.75
	Label Smooth	65.71(+0.19)	56.29(+0.54)
	Zipf's LS	**66.03 (+0.51)**	**56.45 (+0.70)**

4.3   Large-scale and Fine-grained Image Classification Tasks

Comparison with state-of-the-art methods. Our method is one-pass with almost zero extra computational or memory cost. Label smoothing and TF-KD are the two most related works with ours as shown in Fig. 1 and Table 1. And CS-KD is the most superior method on small-scale datasets besides ours as shown in Table 2. So we further compare our method with CS-KD, label smoothing and TF-KD on large-scale and fine-grained datasets. As shown in Table 3, our method shows much more superior performance while label smoothing and CS-KD methods have already improved baseline with significant margins. For instance, we surpass the second-best method by 0.47% and 0.59% respectively on ImageNet and INAT21 based on ResNet50.

Improvements on various architectures. We evaluate our method on various network architectures on ImageNet and INAT21. Not only the widely used families of ResNet [9] and ResNeXt [26] are considered, but also the lighter

architectures (such as MobileNetV2 [22]) are evaluated. Table 4 shows our significant improvements compared with vanilla cross-entropy training on various network architectures based on ImageNet and INAT21 datasets. For instance, our method boosts baseline by 0.75% and 2.85% respectively on ImageNet and INAT21 with ResNet101.

Table 5: Dense classification ranking v.s. logits-based ranking on TinyImageNet with ResNet18. CE indicates the standard cross-entropy loss. LR indicates that using the logits ranks. Dense1 indicates that using the last dense feature map of the last stage. Dense2 indicates that using the last dense feature map of the penultimate stage

Method	CE	LR	Dense1	Dense2	Top-1 acc(%)(gain)
Vanilla	✓				56.41
Logits-based	✓	✓			(+1.70)
Voting-based	✓	✓	✓		(+2.40)
Voting-based	✓	✓	✓	✓	**(+2.84)**

4.4   Ablation Study

Dense classification ranking v.s. Logits-based ranking. We introduced two ranking metrics in section 3.2, logits-based ranking and dense classification ranking. While logits-based ranking indeed boosts performance compared to the baseline (1.7% as shown in the second row of Table 5), we still find our strategy, the dense classification ranking, is necessary for optimal performance. When we replace the rank by the dense votes from the last few feature maps, the accuracy improvement goes to as high as 2.84% (last row in Table 5).

Comparison of different distributions. We constraint the logits rank or dense vote rank to obey Zipf's law due to the discovery that the output of deep neural network trained for a classification task follows the law as well. To demonstrate Zipf's priority, we conduct constant style, random style and decay style distributions on various datasets. As shown in Table 6, Zipf makes the best performance among these distributions. It's worth noting that, although inferior to Zipf's distribution, constant distribution also achieves satisfying performance which benefits from regularizing only non-target classes different from normal label smoothing. We speculate that label smoothing coupled with the target class that might hurt performance.

5   Discussion

Zipf's soft labels results in more reasonable predictions. Fig. 4 illustrates the top-5 predictions of our proposed method compared with the baseline method. The images are sampled from ImageNet and INAT21 respectively. Our method makes more reasonable predictions. Not only the top-1 prediction is correct, but also more similar concepts arise in the top-5 prediction. This results
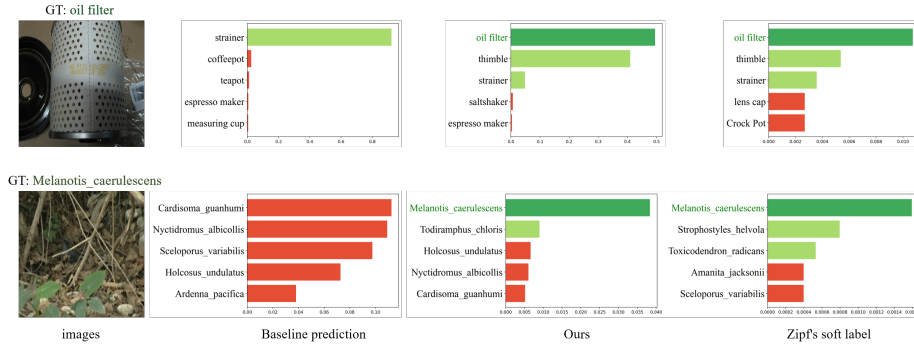
Fig. 4: Comparison of top-5 predictions visualization. The dark green, light green, and red colors denote ground-truth, similar and irrelevant categories respectively. More informative supervision and prediction are gained since categories that are more similar to ground truth arise top in Zipf's soft label (thimble and strainer vs oil filter).

from more informative soft labels from Zipf's LS, which provides meaningful representations as knowledge distillation for the network to better grasp the concept of similar categories.

Table 6: Comparison of the top-1 accuracy (%) of different distributions and label smoothing. Experiments are conducted on TinyImageNet, ImageNet and INAT21. Zipf makes the best performance among these distributions

Distribution	TinyImageNet	ImageNet	INAT21
Vanilla	56.41	76.48	62.43
LS	56.89	76.67	65.16
Constant	58.76	77.09	65.86
Random_Uniform	58.24	76.89	65.61
Random_Pareto	58.52	76.61	65.9
Linear_Decay	58.39	76.87	65.86
Zipf	**59.25**	**77.25**	**66.04**

Zipf's Label Smoothing achieves better representation learning for generalization. We compare our Zipf's LS techniques with cross-entropy training and uniform label smoothing training on the TinyImageNet dataset. As shown in Fig. 5, the intra-class distance in feature space learned from Zipf's label smoothing is more compact and the inter-class distance is more separate. Zipf's Label Smoothing achieves better representation learning for generalization. More discussion for generalization is shown in supplements.

Non-target class dense classification ranking design. To better rank the classes in the soft label, we exploit dense classification ranking instead of logits-based ranking. Thus, larger objects are favored and small object classification performance might degrade. In our design, the target class is excluded in L_{Zipf}
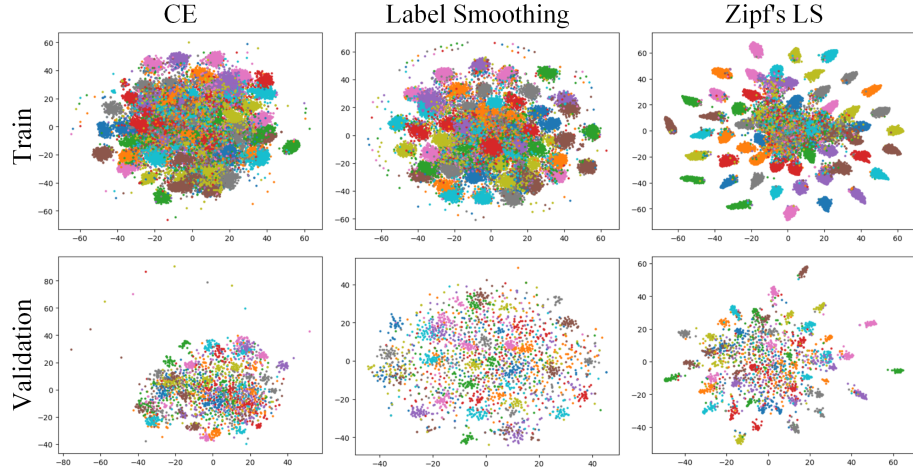
Fig. 5: T-SNE [17] visualization on 50 random sampled classes of TinyImageNet for CE, Label Smoothing and Zipf's label smoothing.
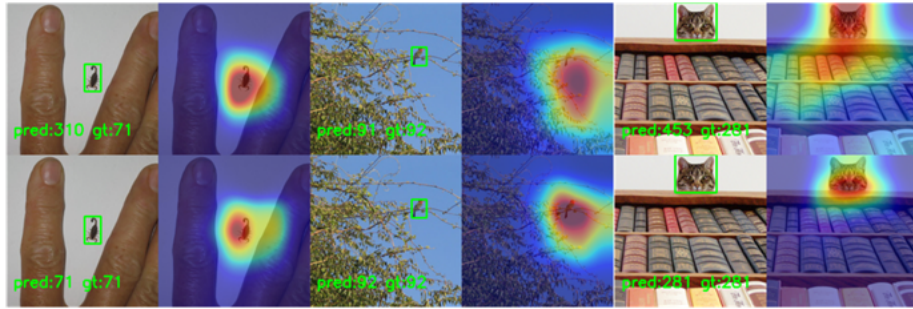

Fig. 6: Comparison of small objects results between CE(above) and Zipf's LS(below).

and included in L_{CE} only, ensuring the target class provides the correct gradient regardless of the object size. To verify the small object performance, we collect the five smallest objects from each class[3], Zipf's LS still beat CE(69.25% vs 67.72%). The cam visualization for the small object case is shown in Fig 6

Limitation. Zipf's LS does not help in binary classification cases, since the L_{zipf} only considers non-target class and would always be zero. Further, as mentioned in the method section, we exploit dense classification ranking to get more reliable ranking information than logit-based ranking, which is only available in image data but not in modals such as speech and language. To make Zipf's LS work in multi-modal data is considered as future work.

---

[3] https://image-net.org/data/bboxes_annotations.tar.gz

6   Conclusion

In this work, we propose an efficient and effective one pass self-distillation method named Zipf's Label Smoothing, which not only generates soft-label supervision in a teacher-free manner as efficient as label smoothing but also generates non-uniform ones as informative as more expensive self distillation approaches. Zipf's Label Smoothing consistently performs better than the uniform label smoothing method and other parameters-free one pass self-distillation methods, it could be one of the plug-and-play self-distillation techniques in your deep learning toolbox.

References

1. Allen-Zhu, Z., Li, Y.: Towards understanding ensemble, knowledge distillation and self-distillation in deep learning. arXiv preprint arXiv:2012.09816 (2020)
2. Bagherinezhad, H., Horton, M., Rastegari, M., Farhadi, A.: Label refinery: Improving imagenet classification through label progression. arXiv preprint arXiv:1805.02641 (2018)
3. Beyer, L., Zhai, X., Royer, A., Markeeva, L., Anil, R., Kolesnikov, A.: Knowledge distillation: A good teacher is patient and consistent. arXiv preprint arXiv:2106.05237 (2021)
4. Chen, D., Mei, J.P., Wang, C., Feng, Y., Chen, C.: Online knowledge distillation with diverse peers. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 3430–3437 (2020)
5. Cho, J.H., Hariharan, B.: On the efficacy of knowledge distillation. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4794–4802 (2019)
6. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
7. Furlanello, T., Lipton, Z., Tschannen, M., Itti, L., Anandkumar, A.: Born again neural networks. In: International Conference on Machine Learning. pp. 1607–1616. PMLR (2018)
8. Guo, Q., Wang, X., Wu, Y., Yu, Z., Liang, D., Hu, X., Luo, P.: Online knowledge distillation via collaborative learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11020–11029 (2020)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition (2015)
10. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification (2015)
11. He, T., Zhang, Z., Zhang, H., Zhang, Z., Xie, J., Li, M.: Bag of tricks for image classification with convolutional neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 558–567 (2019)
12. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
13. Ji, M., Shin, S., Hwang, S., Park, G., Moon, I.C.: Refine myself by teaching myself: Feature refinement via self-knowledge distillation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10664–10673 (2021)
14. Kim, K., Ji, B., Yoon, D., Hwang, S.: Self-knowledge distillation with progressive refinement of targets. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6567–6576 (2021)
15. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Tech. Rep. 0, University of Toronto, Toronto, Ontario (2009)
16. Ma, H., Chen, T., Hu, T.K., You, C., Xie, X., Wang, Z.: Undistillable: Making a nasty teacher that cannot teach students. ArXiv **abs/2105.07381** (2021)
17. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(11) (2008)
18. Mirzadeh, S.I., Farajtabar, M., Li, A., Levine, N., Matsukawa, A., Ghasemzadeh, H.: Improved knowledge distillation via teacher assistant. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 5191–5198 (2020)

19. Müller, R., Kornblith, S., Hinton, G.: When does label smoothing help? arXiv preprint arXiv:1906.02629 (2019)
20. Park, D.Y., Cha, M.H., Kim, D., Han, B., et al.: Learning student-friendly teacher networks for knowledge distillation. Advances in Neural Information Processing Systems **34** (2021)
21. Powers, D.M.: Applications and explanations of zipf's law. In: New methods in language processing and computational natural language learning (1998)
22. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks (2019)
23. Stanton, S., Izmailov, P., Kirichenko, P., Alemi, A.A., Wilson, A.G.: Does knowledge distillation really work? Advances in Neural Information Processing Systems **34** (2021)
24. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2818–2826 (2016)
25. Van Horn, G., Cole, E., Beery, S., Wilber, K., Belongie, S., Mac Aodha, O.: Benchmarking representation learning for natural world image collections. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12884–12893 (2021)
26. Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks (2017)
27. Xu, T.B., Liu, C.L.: Data-distortion guided self-distillation for deep neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 5565–5572 (2019)
28. Yuan, L., Tay, F.E., Li, G., Wang, T., Feng, J.: Revisiting knowledge distillation via label smoothing regularization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3903–3911 (2020)
29. Yun, S., Park, J., Lee, K., Shin, J.: Regularizing class-wise predictions via self-knowledge distillation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 13876–13885 (2020)
30. Zagoruyko, S., Komodakis, N.: Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. arXiv preprint arXiv:1612.03928 (2016)
31. Zhang, C.B., Jiang, P.T., Hou, Q., Wei, Y., Han, Q., Li, Z., Cheng, M.M.: Delving deep into label smoothing. IEEE Transactions on Image Processing **30**, 5984–5996 (2021)
32. Zhang, L., Bao, C., Ma, K.: Self-distillation: Towards efficient and compact neural networks. IEEE transactions on pattern analysis and machine intelligence **PP** (2021)
33. Zhang, L., Song, J., Gao, A., Chen, J., Bao, C., Ma, K.: Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3713–3722 (2019)
34. Zhu, Y., Wang, Y.: Student customized knowledge distillation: Bridging the gap between student and teacher. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5057–5066 (2021)

Supplementary Materials of
Efficient One Pass Self-distillation with Zipf's Label Smoothing

Jiajun Liang, Linze Li, Zhaodong Bing, Borui Zhao, Yao Tang, Bo Lin, and Haoqiang Fan

MEGVII Technology
{liangjiajun,lilinze,bingzhaodong,zhaoborui,tangyao02,linbo,fhq}@megvii.com

1   Explanation to empirical observation

We find that the Zipf's prior could help generate non-uniform supervision for non-target classes in a one-pass way. In this section, we provide a simple intuition to explain why Zipf's law should occur for predictions from multi-class classification.

We postulate that one main source of the non-zero network predictions is the inevitable non-orthogonality of the inter-class feature vectors as more and more classes are packed into the finite-dimensional feature space. In a simplified model, we assume that the decision vectors corresponding to each class are uniformly distributed on a high-dimensional unit sphere. Then for another random query vector on the sphere, their inner-products with it distribute in the shape of a Gaussian when the dimension is high enough.

We propose a toy experiment to verify that softmax Gaussian logits fit Zipf's law well. As shown in Algorithm 1, first, we sampled random vectors from multivariate normal distribution $\mathcal{N}(\mathbf{0}, I_{1000})$ as logits of different samples. Then logits for each sample are sorted and applied with softmax to get probabilities. At last, we average the sorted probabilities across all samples and plot the probabilities-
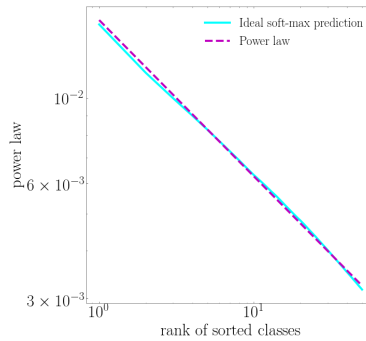

Fig. 1: softmax Gaussian logits fit Zipf's law well

rank relation for the top 32 categories in log-log space. It could be seen that a straight line pattern shows up in Figure 1.

Algorithm 1: simulation of ranking softmax Gaussian logits

```
# generate Gaussian logits,1000 samples, 1000 classes
logits = np.random.randn(1000, 1000)
# sort in class dimension
sorted_logits = np.sort(logits, axis=1)
# probability predictions by applying soft-max on the logits
sorted_preds = np.exp(sorted_logits) /
 np.sum(np.exp(sorted_logits),axis=1)[:,None]
# averaged across samples
mean_sorted_preds = np.mean(sorted_preds, axis=0)
# top 32 ranks considered
top32_sorted_preds = mean_sorted_preds[::-1][:32]
```

We also compare several most frequently-used distributions of long-tail shapes (Zipf's law, exponential and log-normal) to fit the empirical softmax scores, as shown in Table 1. All parameters of distributions are estimated by most-likelihood estimation. Common statistical test metrics such as R square are measured. Zipf's law outperforms the other two in all kinds of metrics.

Table 1: Statistical test of how well different distributions fit on empirical averaged predictions on INAT-21. The top 50 categories are considered. For tests such as Chisquare and Kolmogorov–Smirnov which heavily rely on the amount of the samples, we sample 10^5 instances from the empirical distribution. D is the Kolmogorov–Smirnov statistic and p is the p-value. Zipf's law outperforms the other two by all kinds of metrics.

Metric	Zipf's law	Exponential	Log-normal
R^2	0.99992	0.6768	0.9672
Kullback–Leibler divergence	0.0000667	0.315	0.0219
Jensen–Shannon divergence	0.0000167	0.063	0.00544
Chisquare	13.3	451677	4499
Kolmogorov–Smirnov	D=0.00278 p=0.42	D=0.265 p=0.0	D=0.0823 p=0.0

2   More Experiment Details and Discussion

2.1   Hyperparameters

Hyperparameters setting rules. Table 2 shows the detail of hyperparameters settings for different tasks. α controls the decay shape of Zipf's distribution,

Table 2: The detail of hyper parameters for different datasets.

Dataset	λ	α	dense layer	β
CIFAR100	0.1	1.0	2	0.1
TinyImageNet	1.0	1.0	2	0.5
ImageNet	0.1	1.0	1	/
INAT21	1.0	1.0	1	/

and is set to 1.0 in all tasks. λ controls the regularization strength, which is set to 0.1 for CIFAR100 and ImageNet, and 1.0 for TinyImageNet and NAT21. For datasets with large-resolution inputs, such as ImageNet and INAT, using the final dense feature maps would be sufficient, and no more dense layers are required. For low-resolution tasks such as CIFAR100 and TinyImageNet, we use one more dense layer to get enough votes for dense ranking. In this case, we need β to weigh the cross-entropy loss for learning the extra classifier. β is set to 0.1 and 0.5 respectively on CIFAR100 and TinyImageNet.

Hyperparameters ablation study. 1) α is not sensitive where $\alpha \in [0.5, 1.5]$. 2) λ is to control regularization strength and is positively correlated with the train/val acc gap. For tasks that are prone to overfitting(TinyImageNet and INAT whose train/val gap are 35% and 25%), λ is 1.0. For tasks that are less overfitting(ImageNet train/val gap is 4%), λ is 0.1. 3) β is optional and only recommended for small resolution tasks. It should be less than 0.5 to avoid shadow learning of deeper layers. See Table 3 for details.

Table 3: Ablation study of hyperparameters α,λ and β

α	0.1	0.5	1.0	1.5	2.0
CIFAR100	77.21±0.29	77.26±0.13	**77.38±0.32**	77.12±0.24	76.45±0.12
TinyImageNet	58.85±0.16	59.06±0.21	**59.25±0.20**	58.64±0.18	53.35±0.41

λ	0.01	0.1	0.5	1.0	1.5
CIFAR100	76.59±0.15	**77.38±0.32**	76.62±0.24	76.79±0.04	76.92±0.17
TinyImageNet	56.86±0.36	57.65±0.01	58.41±0.17	**59.25±0.20**	58.03±0.27

β(optional)	0.05	0.1	0.3	0.5	0.7
CIFAR100	77.24±0.22	**77.38±0.32**	76.75±0.31	76.39±0.16	76.49±0.24
TinyImageNet	58.71±0.17	58.77±0.18	59.48±0.31	**59.25±0.20**	59.00±0.12

2.2   SNR of Ranking

Ranking the classes accurately is a key factor to generate proper Zipf's law distribution for the sample. In the method section, we propose a finer ranking method named dense classification ranking which exploits spatial classification results from the last few feature maps. A consequence of this voting-based method is
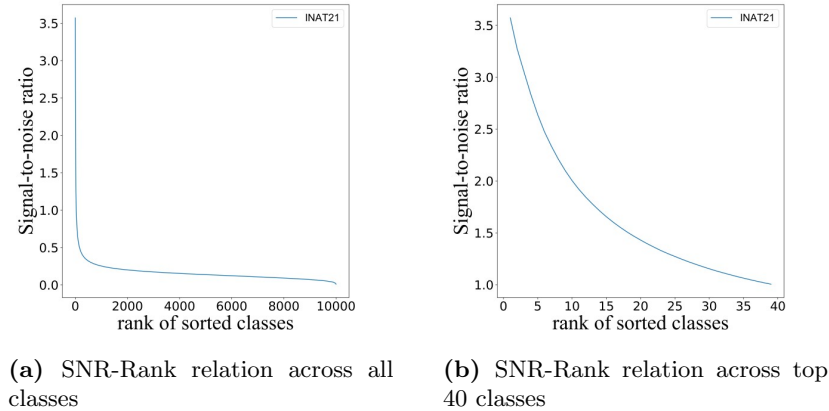
(a) SNR-Rank relation across all classes

(b) SNR-Rank relation across top 40 classes

Fig. 2: SNR-Rank relation plot for trained ResNet-50 model in INAT21 dataset. It can be seen that only the top 40 rankings have SNR larger than 1, which makes it hard to give reliable ranks for tailing class on the fly.

that we have to clip the Zipf's values to a uniform one after a certain rank, as they would not receive sufficient votes to be distinguished individually. To illustrate that ranking only a few top non-target classes is sufficient, we study the signal-to-noise ratio of rankings. The signal and noise of specific rank r are calculated as the average and standard deviation among r-th probabilities from different sorted samples respectively. As shown in Figure.2, we plot the SNR-rank curve on the INAT21 dataset, only the top 40 out of 10000 ranks whose SNR is larger than one. It's a good trade-off to just give power-law decayed probabilities to the top-ranking class since the SNR of tailing ranks is too low to give reliable ranks.

2.3 More Zipf's Soft Label Examples

Figure 4 illustrates more results of the top-5 predictions of our proposed method compared with the baseline method. The top three rows are sampled from ImageNet while the bottom three rows are sampled from INAT21. It can be seen that: (1) There are several categories similar to the target class shown up in Zipf's soft labels, which provide meaningful label representations for the network to better grasp the concept of the target class. (2) Fine-grained categories of the target class emerge in Zipf's soft labels, which can provide similar "dark knowledge" as knowledge distillation.
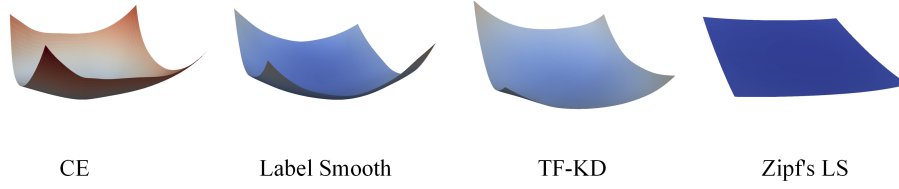
CE   Label Smooth   TF-KD   Zipf's LS

Fig. 3: Comparison of loss landscape with efficient teacher-free methods

2.4   Generalization: Performance on downstream tasks

To measure the power of generalization of Zipf's LS, we conducted the transfer learning task by fine-tuning ImageNet pre-trained models on MS-COCO, as shown in Table 4. Besides, we visualize loss landscapes [**?**] of several efficient teacher-free methods(see Fig 3), Zipf's LS achieves more flat convergence, which is a possible hint for better generalization [**?**].

Table 4: ImageNet pretrained ResNet50 for object detection

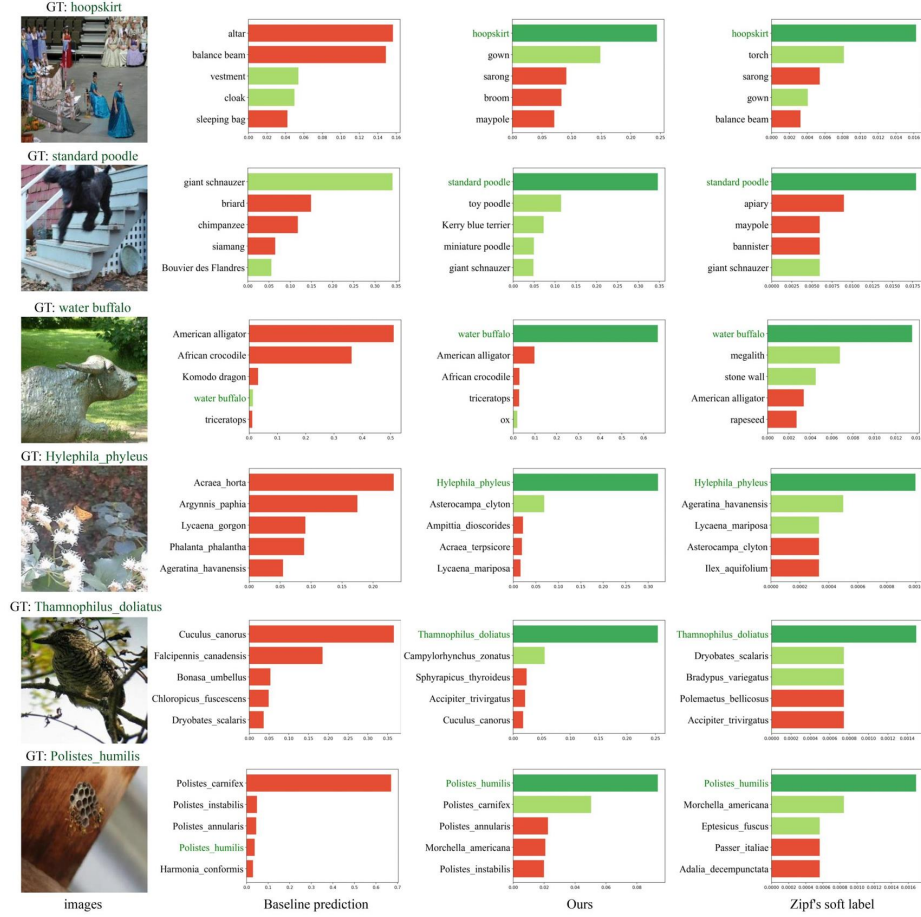| Method	Vanilla(CE)	TF-KD	PS-KD	Zipf's LS(Ours)
AP	36.4%	36.4%	36.5%	**36.6%**
AP@0.5	58.3%	56.7%	56.7%	**58.8%**

Fig. 4: Top-5 predictions visualization of our proposed method (Zipf's label smoothing) compared with the baseline method (cross-entropy). The dark green, light green and red denote ground-truth, similar and irrelevant categories respectively. "GT" denotes the ground truth label and thus the hard label of the baseline method. The baseline prediction is acquired under the supervision of the hard label and misclassified on the samples. Our method exploits target-relevant categories to better represent the image, and obtains better results.