Knowledge Distillation Dealing with Sample-wise Long-tail Problem

Tao Yu[1,2], Xu Zhao[1,3(✉)], Yongqi An[1,2], Ming Tang[1,2], and Jinqiao Wang[1,2,3]

[1] Institute of Automation, Chinese Academy of Sciences
yutao2022@ia.ac.cn
{xu.zhao,yongqi.an,tangm,jqwang}@nlpr.ia.ac.cn
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences
[3] Wuhan AI Research

Abstract. We discover that while knowledge distillation improves the overall performance of student models, the performance improvement for some samples in the tail is limited, which is a rarely addressed issue. These tail samples can lead to poor learning of the teacher's feature distribution in the corresponding regions of the feature space, thereby limiting the alignment between the student and the teacher. Since tail samples often lack clear label definitions in many tasks, we identify them by analyzing the average feature similarity from the teacher model. To improve knowledge distillation, we propose a Sample-wise Re-weighting (SRW) method, assigning different loss function weights to samples based on their average similarity. Experimental results show that our method enhances the performance of student models across different tasks and can be combined with various knowledge distillation methods. Additionally, our approach demonstrates advantages in foundational models such as Segmentation Anything Models (SAM) and Contrastive Language–Image Pretraining (CLIP) models.

Keywords: Knowledge Distillation · Vision-Language Models · Segmentation.

1 Introduction

Knowledge distillation [7] is a widely used technique in the field of deep learning. This technique involves a student model learning from the outputs or features of a teacher model. It is commonly applied in model compression as well as in aligning capabilities and outputs between models. Since Hinton *et al.* proposed the original knowledge distillation method, many researchers have continuously improved both logit distillation [11, 21, 37, 39, 30] and feature distillation [1, 26, 18, 6] methods, significantly enhancing the performance of student models.

All knowledge distillation methods can be unified under the problem of learning the teacher's feature distribution. Although logit distillation focuses on replicating the teacher's output, it shares the property of continuous variation with feature distributions. Thus, logit distillation can be considered a specialized form

of feature distribution learning. Generally, the more accurately the student model learns the teacher's feature distribution, the better its performance will be.

To improve the student's learning of the teacher's feature distribution, many approaches have been proposed from various perspectives. For example, DKD [39] decomposes the original logit distillation into learning binary probabilities of the target class and all the other non-target classes and learning the probability distribution within non-target classes. DKD finds that the latter is suppressed by the original distillation method, and thus, it strengthens the learning of it. [34] finds that the high-frequency information in the feature distribution is poorly learned, so it transforms the features into the frequency domain using DFT and reinforces the learning of high-frequency information.

However, the aforementioned studies have overlooked analyzing how to improve the learning of teacher's feature distribution from the perspective of samples. Current research indicates that in both pre-trained models and downstream tasks, there are not only class-wise long-tail problems but also sample-wise ones. In knowledge distillation, the goal is to learn the teacher's feature distribution. However, the student learns from discrete feature points, and regions with denser sampling in the feature space are learned better, while sparser regions are learned less effectively.

Based on the above analysis, we propose a very simple yet effective method to alleviate the sample-wise long-tail problem in the learning of teacher's feature distribution. Since the features sampled in sparse regions have lower similarity with other features, we calculate the average similarity of the teacher's feature with those of other samples to determine if the sample's feature is in a sparse region. This allows us to identify whether the corresponding sample is a tail sample. Inspired by the commonly used re-weighting methods for addressing the class-wise long-tail problem, we propose a **S**ample-wise **R**e-**W**eighting (**SRW**) method. This method assigns greater weights to the loss functions of tail samples based on the calculated average similarity, thus enhancing the learning of the corresponding samples.

Experimental results show that our method can improve the accuracy of the original knowledge distillation method on CIFAR-100 [14] and ImageNet-1K datasets [27]. Furthermore, our approach can be combined with other more advanced knowledge distillation methods. The pre-training data for many foundational models also exhibits imbalanced samples. We have also attempted to combine our method with existing distillation methods on several foundational models, such as Contrastive Language–Image Pretraining (CLIP) [24] and Segment Anything Model (SAM) [13]. Notably, our approach can be combined with MobileSAM [36] to improve the segmentation performance of lightweight SAMs.

Our main contribution is three-fold:

- We unify knowledge distillation as a problem of learning the teacher model's feature distribution and identify a sample-wise long-tail problem in the process of learning this feature distribution.
- To address the aforementioned issue, we draw inspiration from the re-weighting method commonly used in class-wise long-tail problems and propose a Sample-

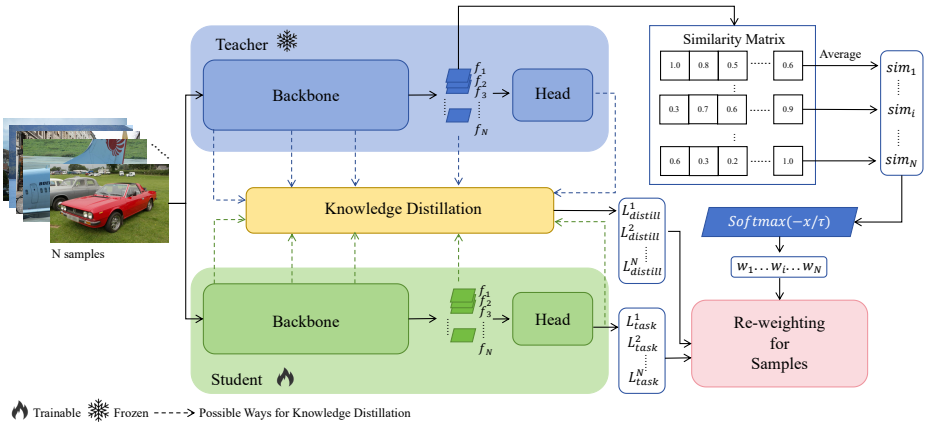

Fig. 1. Overview. We calculate the average similarity of the teacher's feature with those of other samples to determine if the sample's feature is in a sparse region of the feature space. This allows us to identify whether the corresponding sample is a tail sample. Further, we propose the **S**ample-wise **R**e-**W**eighting (**SRW**) method. This method assigns greater weights to the loss functions of tail samples based on the calculated average similarity, thus enhancing the learning of the corresponding samples.

wise Re-Weighting (SRW) method to enhance the learning of the corresponding feature distribution.
– Our method can be combined with various existing knowledge distillation methods to improve their performance in tasks such as image classification. It is also effective in model compression for foundational models like CLIP and SAM.

2 Related Work

2.1 Knowledge Distillation

A decade ago, Hinton *et al.* proposed the knowledge distillation method [7]. The original method transfers dark knowledge to the student model by minimizing the Kullback-Leibler divergence between the student model's output and the teacher's soft labels. Subsequent improvement methods have attempted to transfer different forms of knowledge to the student model, mainly in three forms: logits [11, 21, 37, 39, 30], features [1, 26, 18, 6] , and relational information [9, 22, 23]. Logit distillation is an enhancement of the original method, obtaining knowledge from the teacher model's output. Feature distillation involves learning the intermediate layer features of the teacher model. Relational distillation focuses on learning the relationships between samples or between different layers' features, thereby learning structured knowledge. Knowledge distillation can

be applied not only to classification tasks but also to object detection [15], segmentation [20], and even multi-modal tasks [10], making it a highly versatile method.

However, previous works have rarely addressed the sample-wise long-tail problem in knowledge distillation. Through analysis, we unify distillation as the learning of the teacher model's feature distribution and find that some regions of the feature space have fewer sampled data, leading to poorer learning of corresponding feature distribution. We propose a simple and effective method to alleviate this issue.

2.2 Re-weighting Method in Long-tail Learning

Our method is closely related to research on long-tail learning. The long-tail learning problem refers to the situation where the dataset has an imbalanced distribution of categories, with the majority of samples concentrated in a few categories (head) and a large number of categories with very few samples (tail). This imbalanced distribution leads to the model performing well on head categories during training but underperforming on tail categories, thereby affecting overall performance. Methods to address this issue include re-sampling [12], re-weighting [19, 2, 25, 35], two-stage training [16, 5] and multi-expert methods [40, 31]. Our approach draws on the re-weighting method.

Re-weighting methods typically attempt to assign different loss weights to different classes to enhance the learning of tail categories. For example, the focal loss [19] increases the loss weight of poorly classified samples by adding a weighting factor to the cross-entropy (CE) loss function. CBCE loss [2] calculates the weighting based on the effective number of samples in different classes rather than the actual number. BSCE loss [25] finds that the softmax function gives biased gradient estimates under long-tail settings, so it proposes an unbiased extension of Softmax to adapt to the label distribution shift between training and testing. CDT [35] introduces a temperature coefficient related to the number of category samples into the softmax function, simulating feature deviation during training to amplify the decision values of tail classes.

Although our method draws on re-weighting methods, it essentially addresses the issue of imbalanced samples in knowledge distillation rather than the long-tail problem of classes. Our method assigns different weights to each sample individually to enhance the learning of teacher model's feature distribution.

3 Method

3.1 Basic Method of Knowledge Distillation

When training with knowledge distillation method, the student model needs to minimize both the task loss $\mathcal{L}_{task}$ and the distillation loss $\mathcal{L}_{distill}$, which measures the discrepancy with the teacher. Given a batch of samples $\{(x_i, y_i)\}_{i=1}^{N}$, the loss function for logit distillation generally uses Kullback-Leibler (KL) divergence.

The specific formula for the distillation loss of a single sample x_i is expressed as follows:

$$\mathcal{L}_{distill}^{i} = T^2 \mathrm{KL}\left(\sigma\left(q_t^i/T\right), \sigma\left(q_s^i/T\right)\right) \tag{1}$$

where q_t and q_s denote the logits produced by teacher and student, σ is the softmax function, and T is the temperature to smooth the distribution of model outputs.

The commonly used loss function for feature distillation is the Mean Squared Error (MSE):

$$\mathcal{L}_{distill}^{i} = \mathrm{MSE}\left(F_t^i, F_s^i\right) \tag{2}$$

where F_t and F_s denote features of the teacher and student model.

For more advanced distillation methods, $\mathcal{L}_{distill}^{i}$ can take on more complex forms. Overall, the distillation loss function employed during training can be represented as the mean of the loss functions computed for each individual sample within a given batch:

$$\mathcal{L}_{distill} = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}_{distill}^{i} \tag{3}$$

The aforementioned formula assigns a uniform weight of $1/N$ to the loss function of each sample. However, due to the imbalanced nature of the dataset, tail samples make up a smaller proportion of the loss function. As a result, treating all samples equally can lead to inadequate learning for these tail samples. To mitigate this issue, it is necessary to assign greater weights to the tail samples within the loss function.

3.2 Identifying Tail Samples

As described in Section 1, knowledge distillation can be seen as learning the teacher's feature distribution. However, the sampling of distribution in the feature space is imbalanced: certain regions have dense feature points while others are sparse. Regions with dense sampling have better-learned feature distributions. To improve the learning of feature distributions in sparse regions, the first step is to identify the tail samples. Given that tail samples lack explicit label definitions and estimating the number of sampled feature points in specific regions of the feature space is challenging, we design a simple and effective approximation method.

Specifically, we propose calculating the average similarity with all samples to identify samples in the tail. For a batch of samples with a batch size of N, the average similarity of each sample is defined by Eq. 4:

$$\mathrm{avesim}_i = \frac{1}{N}\sum_{j=1}^{N} \cos(f_t^i, f_t^j) = \frac{1}{N}\sum_{j=1}^{N} \frac{f_t^i \cdot f_t^j}{\|f_t^i\|\|f_t^j\|} \tag{4}$$

where f_t represents the features to be learned from the teacher model.

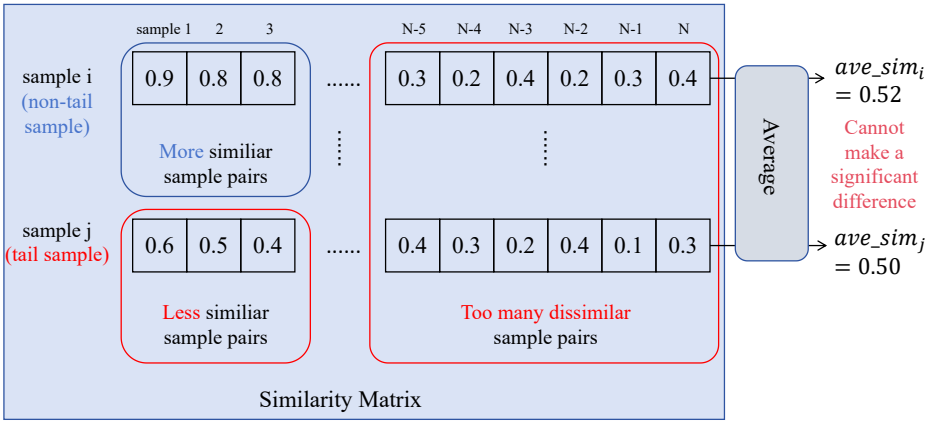

Fig. 2. Analysis why a normalization method based on relative values is necessary. In all pairs of samples, dissimilar pairs are the majority, causing the average similarity between tail samples and non-tail samples to be indistinguishable. Using a calculation method based on relative values can better differentiate between them.

3.3 Sample-wise Re-weighting (SRW)

When calculating the normalized weight of the loss function for each sample based on average similarity, using relative rather than actual similarity values is essential. This is because, for each sample, the dissimilar samples make up the majority of the dataset, leading to the average similarity of tail and non-tail samples being quite similar. Weights calculated based on absolute values cannot differentiate effectively, and thus, cannot enhance the learning of tail samples, as analyzed in Fig. 2.

Among the functions that meet the above requirements, the softmax function is the most commonly used. The property $\sigma(x) = \sigma(x - a)$ ensures that the relative values determine the output. Therefore, the calculation of weights is defined as Eq. 5:

$$w_i = \frac{e^{(-\frac{\text{avesim}_i}{\tau})}}{\sum_{j=1}^{N} e^{(-\frac{\text{avesim}_j}{\tau})}} \tag{5}$$

The negative sign in the above equation is used to assign greater weight to tail samples (those with low average similarity). τ is a newly introduced hyper-parameter that controls the weighting strength. Subsequent experimental results will demonstrate that this is a very important hyperparameter, influencing the extent to which the accuracy of the student model is improved.

Based on the weights calculated earlier, $\mathcal{L}_{distill}$ can be modified to a weighted form. For a batch of samples, $\mathcal{L}_{reweight}$ is given by:

$$\mathcal{L}_{reweight} = \sum_{i=1}^{N} w_i L_{distill}^{i} \tag{6}$$

4 Experiment

4.1 Results on Image Classification

Datasets. We conduct experiments using two classic image classification datasets: CIFAR-100 [14] and ImageNet-1K [27]. The CIFAR-100 dataset contains 100 categories, with image sizes of 32x32. The training set and validation set contain 50k and 10k images, respectively. ImageNet-1K is a large-scale dataset with 1000 categories, and the training and validation sets contain 1.28 million and 50k images, respectively.

Implementations. We follow the most commonly used experimental settings in the field of knowledge distillation research. Model architectures include classic structures such as VGG [29], ResNet [4], and MobileNet [8]. The settings of teacher and student include using the same architecture and different architectures. We searched for and set the optimal hyperparameter τ for each experiment as thoroughly as possible. Other hyperparameters, such as learning rate, batch size, distillation temperature, and those introduced by various distillation methods, are set to the default values used in previous works. More details on hyperparameter settings can be found in [39] and [38].

Results. Tab. 1 shows the knowledge distillation accuracy results on the CIFAR-100 dataset when the teacher and student models have the same architecture. To demonstrate that our method can be combined with various other distillation methods to further improve accuracy, we present the improvement in accuracy (Δ) when combined with KD [7], FitNet [26], and DKD [39] methods. KD and FitNet are classic logit and feature distillation methods, respectively, whereas DKD is a more advanced and widely adopted knowledge distillation approach from recent years. Experimental results show that our method can be combined with various existing knowledge distillation methods to improve accuracy. This fully demonstrates the effectiveness of our approach.

Table 1. Knowledge distillation results on CIFAR-100 datasets for models with the **same architecture**. Δ represents the improvement in accuracy compared to original methods.

Teacher		Student		KD			FitNet			DKD		
				vanilla	+SRW	Δ	vanilla	+SRW	Δ	vanilla	+SRW	Δ
ResNet56	72.34	ResNet20	69.06	70.66	71.16	0.5	69.21	69.32	0.11	71.97	72.19	0.22
ResNet110	74.31	ResNet32	71.14	73.08	73.66	0.58	71.06	71.09	0.03	74.11	74.2	0.09
ResNet32×4	79.42	ResNet8×4	72.5	73.33	73.93	0.6	73.5	73.75	0.25	76.32	76.53	0.21
VGG13	74.64	VGG8	70.36	72.98	73.88	0.9	71.02	71.18	0.16	74.68	74.99	0.31

Tab. 2 shows the knowledge distillation accuracy results on the CIFAR-100 dataset when the teacher and student models have different architectures. Similar to the experiments in Table 1, in this experiment, we combine our method with various other methods to demonstrate its effectiveness in heterogeneous settings.

Tab. 3 reports the Top-1 accuracy results on the ImageNet-1K dataset. The experiments show that our method can also improve the accuracy of existing

Table 2. Knowledge distillation results on CIFAR-100 datasets for models with **different architectures**. Δ represents the improvement in accuracy compared to original methods.

Teacher		Student		KD			DKD		
				vanilla	+SRW	Δ	vanilla	+SRW	Δ
WRN-40-2	75.61	ShuffleNet-V1	70.5	74.83	76.05	1.22	76.7	77.32	0.62
VGG13	74.64	MobileNet-V2	64.6	67.37	68.53	1.16	69.71	70.03	0.32
ResNet50	79.34	MobileNet-V2	64.6	67.35	68.84	1.49	70.35	70.77	0.42

knowledge distillation methods on this dataset, demonstrating the effectiveness of our approach on more challenging datasets.

Table 3. Knowledge distillation results on the ImageNet-1K dataset. Δ represents the improvement in Top-1 accuracy compared to original methods.

Teacher		Student		KD		
				vanilla	+SRW	Δ
ResNet34	73.31	ResNet18	69.75	70.66	70.93	0.27
ResNet50	76.16	MobileNet-V1	68.87	68.58	69.08	0.50

Improvement on samples in the tail. To further demonstrate the effectiveness of our method, Tab. 4 showcases the substantial improvements achieved on the tail samples. According to our analysis, tail samples are those that exhibit a relatively low average similarity to other samples. Given the computational burden of calculating average similarity across the entire validation set, we design our evaluation method to operate within batches. Within each batch, samples are ranked by their average similarity in descending order and those ranked lower are considered tail samples. During evaluation, the batch size is set to 64. The results indicate that our method significantly enhances the accuracy for tail samples without compromising the accuracy of non-tail samples, thereby improving the overall performance of the model.

Table 4. The improvement in accuracy across different samples when combining the KD method with SRW. Within this experiment, the average similarity is calculated within the same batch (with a batch size of 64), and those samples ranked lower are considered tail samples. The experiment is conducted on CIFAR-100.

Teacher	Student	Top 1-16	Top 17-32	Top 33-48	Top 49-64
ResNet56	ResNet20	0.01	0.30	0.50	1.19
ResNet32×4	ResNet8×4	0.04	0.33	0.82	1.21

4.2 Results on CLIP

Introduction to CLIP. CLIP [24], or Contrastive Language–Image Pretraining, is a model released by OpenAI that aligns image and text features through a contrastive learning method. The model consists of two branches: an image encoder and a text encoder. It is widely applied in various multi-modal models.

Datasets. Referring to the experimental setup in [17], we use the CC3M [28] dataset for distilling the CLIP model. This dataset contains image-caption pairs and is used for training and evaluating image captioning systems, with approximately 3.3 million images. Since the original dataset used for CLIP pretraining is very large and we lack sufficient resources for training on such a scale, we use this dataset for the preliminary validation of our method.

Implementations. Referring to the methods in [17] and [3], we only distill the image encoder. In many computer vision tasks, we only need the precomputed text features outputted by the text encoder, such as in zero-shot image classification, where it is not necessary to run the text encoder during each inference. Therefore, compressing the text encoder is not required. Additionally, some multi-modal models also need to align with CLIP's image features [3]. We use TinyViT [33] as the student model and the ViT-base version of CLIP as the teacher model, aligning the outputs of the student model's and teacher model's image encoder using MSE as the distillation loss function. After training, model performance is evaluated by testing zero-shot image classification capability, using the ImageNet-1K dataset mentioned in Section 4.1.

We implement MSE-based knowledge distillation as a baseline. Different distillation methods use the same hyperparameter settings: a learning rate of 0.001, a batch size of 1024, and 10 epochs of training.

Results. Tab. 5 presents the preliminary verification results of our method on CLIP. The results show that the student image encoder pre-trained with our method demonstrates better zero-shot classification capability. Due to the smaller number of parameters in the student model and the smaller scale of the pre-training data, we cannot fully achieve the same accuracy as the original ViT-base. However, there is indeed an imbalance problem in the CLIP pre-training data [32], and we believe that our method can alleviate the long-tail problem on a larger-scale pre-training dataset.

Table 5. Preliminary experimental results of knowledge distillation on the CLIP model. It shows the zero-shot accuracy on ImageNet-1k after using different knowledge distillation methods during pre-training. Δ represents the improvement in accuracy compared to original methods.

	Top1	Top5
MSE	21.80	47.62
+SRW	21.92	47.88
Δ	0.12	0.26

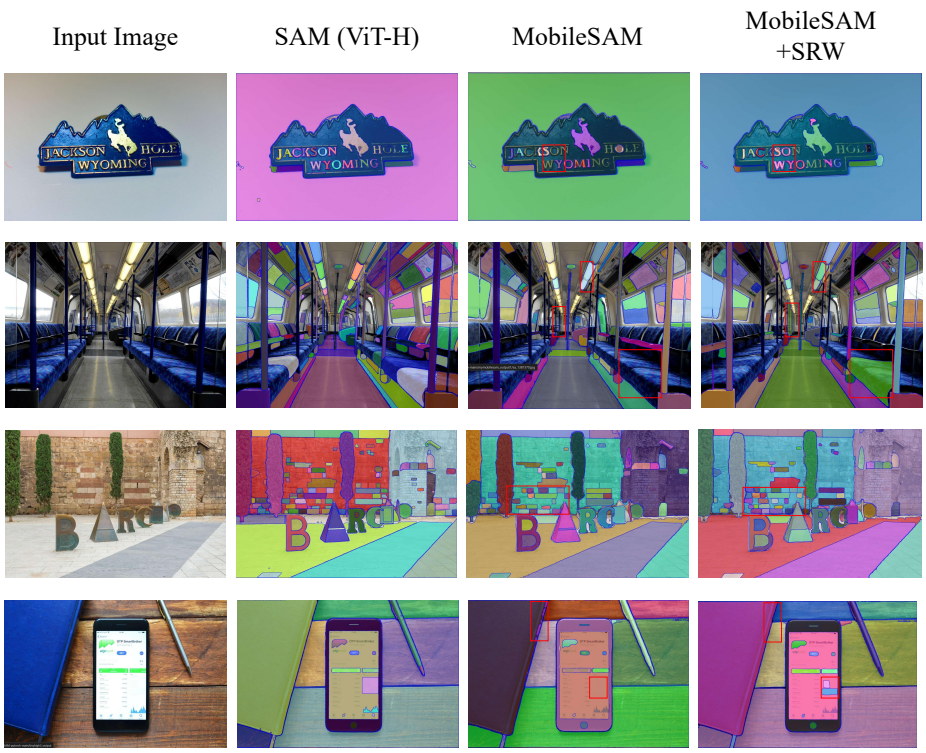

Fig. 3. Visualization of segmentation results. **Red boxes** indicate areas where our method (MobileSAM+SRW) shows noticeable differences in segmentation results compared to MobileSAM.

4.3 Results on Segment Anything

Introduction to SAM. The Segment Anything Model (SAM) [13] can segment corresponding objects based on prompts such as points, boxes, or masks according to user intent. SAM consists of three modules: the image encoder, the prompt encoder, and the mask decoder. The prompt encoder processes the input prompts. These encoded prompts, together with the image features (image embeddings) extracted by the image encoder, are fed into the mask decoder. After applying post-processing steps like Non-Maximum Suppression (NMS) and thresholding, we obtain the mask results.

Datasets. We conduct experiments using the SA-1B dataset [13], which consists of 11 million high-resolution images and 1.1B high-quality segmentation masks. Following the MobileSAM method, we use 1% of the data for training, as lightweight models can converge on this data scale, and additional data does not significantly improve the performance of model.

Implementations. MobileSAM [36] is a classic method for compressing SAM, and we attempted to combine our method with it to enhance its per-

formance. MobileSAM only compresses the image encoder. This method decouples the image encoder and other modules, keeping the original mask decoder and prompt encoder unchanged. It uses the image encoder of the original SAM (ViT-H) as the teacher and Tiny-ViT as the student model. Using MSE as the loss function, the student model learns the image embeddings outputted by the teacher model.

Due to the lack of publicly available specific training details for MobileSAM, including learning rate, optimizer, and specific data used, we reproduce the method ourselves. Our experimental settings are as follows: using the AdamW optimizer, setting the learning rate to 0.002, batch size to 64, and training for 8 epochs.

When applying our method to SAM, we perform weighting at the granularity of image patches. We calculate the average similarity and weight across different image patches within the same image, ensuring that image patches from different images do not affect each other.

Results. In Tab. 6, we use the MIoU metric to show the performance differences of MobileSAM before and after using the re-weighting method. Experiments are conducted with multiple point prompts, such as 5x5, 16x16, or 32x32 points, to evaluate the capability of outputting multiple masks. As the number of input points increases, the model is more likely to segment smaller objects. MIoU is calculated using the prediction results of the original SAM (ViT-H) as the ground truth. The experimental results show that combining our method with MobileSAM can further improve its segmentation quality.

Table 6. Knowledge distillation results on the SAM. MIoU is calculated using the output of the original SAM (ViT-H) as ground truth. The numbers 5x5, 16x16, and 32x32 points refer to the number of point inputs to the prompt encoder to segment multiple objects. Δ represents the improvement in MIoU compared to original methods.

Method	5x5 points	16x16 points	32x32 points
MobileSAM	72.66	73.53	73.42
+SRW	73.33	74.17	74.05
Δ	0.67	0.64	0.63

The MIoU metric may not fully capture the improvement in segmentation quality for finer details, so we also analyze the performance improvement by visualizing the segmentation results. In Fig 3, we use red boxes to highlight areas where there are noticeable differences between our method and MobileSAM in the segmentation results. Overall, after using the re-weighting method, MobileSAM recalls more small targets and refines the edge segmentation.

Visualization of weights. This section presents some interpretability analysis by visualizing which regions of the image our method assigns higher weights to. Fig 4 shows a heatmap that illustrates the weights assigned to different patches within an image in the loss function. The weight increases as the color

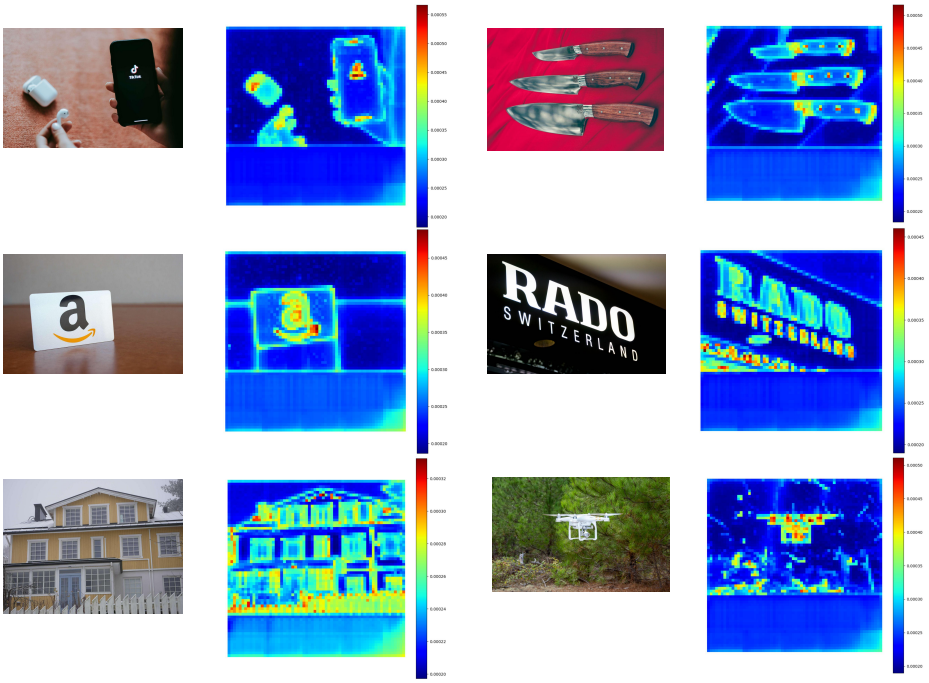

Fig. 4. Visualization of weights for different areas in images. In the heatmap, **red** indicates **higher** weights, and **blue** indicates **lower** weights. The additional parts in the heatmap compared to the original image are patches added by SAM to standardize the number of image patches.

shifts towards red and decreases as it shifts towards blue. The visualization results show that our method tends to assign higher weights to small objects, and within the same object, the edges receive higher weights than the interior. This aligns with the common understanding of tail samples in segmentation tasks: small objects and edges are relatively tail samples. In addition, the image embeddings typically include extra parts compared to the original image. These are additional image patches added by the SAM image encoder to standardize image sizes. From the visualization results, we can see that our method assigns lower weights to these unnecessary image patches as well.

4.4 Ablation Study

In this section, we will demonstrate the effectiveness of each step in our method through ablation experiments, including (1) the effectiveness of the re-weighting method, (2) the setting of the hyperparameter τ, and (3) the effectiveness of the method for calculating normalized weighting values. Due to the large number of experiments involved, it is not possible to present all the ablation study results

in the main text. We primarily showcase the experimental results on the CIFAR-100 dataset.

The effectiveness of the re-weighting method. This section mainly demonstrates the effectiveness of the re-weighting method that assigns higher weights to tail samples. We need to compare this with the method that assigns higher weights to non-tail samples (Reversing). Reversing specifically refers to the softmax function in Equation 5 without the negative sign inside. The results in Tab. 7 show that our method outperforms the Reversing methods, indicating that it is indeed necessary to preferentially enhance the learning of tail samples.

Table 7. The effectiveness of the re-weighting method. Reversing means assigning greater weights to non-tail samples, which is the opposite of our method. This experiment is conducted on the CIFAR-100 dataset.

Teacher	Student	Reversing	Ours
WRN-40-2	ShuffleNet-V1	75.59	76.05
VGG13	MobileNet-V2	67.78	68.53

The setting of the hyperparameter τ. This section focuses on the importance of hyperparameter τ in our approach. Tab. 8 reports the optimal τ we found for different experimental settings. It can be seen that τ has a significant impact on the accuracy of the student model. Setting it too small or too large will result in excessively strong or weak weighting of the tail samples, leading to a decrease in the overall performance of the model.

Table 8. The setting of the hyperparameter τ. This experiment is conducted on the CIFAR-100 dataset.

Teacher	Student	0.5	0.6	0.7	0.8	0.9	1.0
WRN-40-2	ShuffleNet-V1	76.03	75.56	76.05	75.63	75.96	75.71
VGG13	MobileNet-V2	68.13	68.01	68.17	67.88	68.04	68.53

The effectiveness of the method for calculating normalized weighting values. This section discusses the necessity of using the softmax function to calculate normalized weights. We design a normalization method for comparison, as shown in Equation 7. This method involves summing the values and then dividing by the sum to obtain the normalized weights. The subtraction of x_{min} in the formula is to avoid negative numbers. In our experiments, this method is referred to as SumNormalizer. The results in Tab. 9 show that softmax not only has the advantage of determining outputs based on relative values, but its ability to flexibly control weighting strength through τ is also a significant advantage.

$$x_{norm} = \frac{(x - x_{min})}{\sum_j (x_j - x_{min})} \tag{7}$$

Table 9. The effectiveness of the method for calculating normalized weighting values. SumNormalizer is the normalization method we use for comparison, as defined in Eq. 7. This experiment is conducted on the CIFAR-100 dataset.

Teacher	Student	SumNormalizer	Ours
WRN-40-2	ShuffleNet-V1	75.32	76.05
VGG13	MobileNet-V2	68.05	68.53

5 Conclusion and Limitations

Through our analysis, we find that knowledge distillation essentially involves learning the teacher's feature distribution. Generally, the more accurately the feature distribution is learned, the better the performance of the student model. However, this learning process encounters a sample-wise long-tail problem: the sampled features based on the training dataset are imbalanced, with denser regions of sampling resulting in better distribution learning and sparser regions resulting in poorer learning. Therefore, we propose a simple yet effective method to mitigate this issue. By calculating the average similarity between features, we can identify tail samples from sparsely sampled regions and propose a sample-wise Re-weighting (SRW) method to enhance the learning of these tail samples. Experiments on image classification models, CLIP, and SAM all demonstrate that our method can be combined with existing knowledge distillation methods to further improve the performance of the compressed models.

The main limitation of our method is that it only analyzes the sample-wise long-tail problem. Other factors affecting the effectiveness of knowledge distillation algorithms, such as temperature settings and algorithm optimization, are not deeply explored in our study and are not directly related to our topic. However, considering more factors comprehensively could indeed further improve accuracy. Additionally, how this method can be applied to generative models such as large language models and diffusion models is also a question worth further investigation.

Acknowledgments. This study was supported by Strategic Priority Research Program of the Chinese Academy of Sciences under Grant No. XDA27030600, Science and Technology Research and Development Plan of China Railway (No. P2023S001) and National Natural Science Foundation of China (No. 62276260, 62076235, 62206290).

References

1. Chen, P., Liu, S., Zhao, H., Jia, J.: Distilling knowledge via knowledge review. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5008–5017 (2021)
2. Cui, Y., Jia, M., Lin, T.Y., Song, Y., Belongie, S.: Class-balanced loss based on effective number of samples. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9268–9277 (2019)
3. Gu, X., Lin, T.Y., Kuo, W., Cui, Y.: Open-vocabulary object detection via vision and language knowledge distillation. arXiv preprint arXiv:2104.13921 (2021)
4. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
5. He, Y.Y., Wu, J., Wei, X.S.: Distilling virtual examples for long-tailed recognition. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 235–244 (2021)
6. Heo, B., Kim, J., Yun, S., Park, H., Kwak, N., Choi, J.Y.: A comprehensive overhaul of feature distillation. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 1921–1930 (2019)
7. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
8. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
9. Huang, T., You, S., Wang, F., Qian, C., Xu, C.: Knowledge distillation from a stronger teacher. Advances in Neural Information Processing Systems **35**, 33716–33727 (2022)
10. Huang, Y., Wang, Y., Wang, L.: Efficient image and sentence matching. IEEE Transactions on Pattern Analysis and Machine Intelligence **45**(3), 2970–2983 (2022)
11. Jin, Y., Wang, J., Lin, D.: Multi-level logit distillation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 24276–24285 (2023)
12. Kang, B., Xie, S., Rohrbach, M., Yan, Z., Gordo, A., Feng, J., Kalantidis, Y.: Decoupling representation and classifier for long-tailed recognition. arXiv preprint arXiv:1910.09217 (2019)
13. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., et al.: Segment anything. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 4015–4026 (2023)
14. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
15. Li, G., Li, X., Wang, Y., Zhang, S., Wu, Y., Liang, D.: Knowledge distillation for object detection via rank mimicking and prediction-guided feature imitation. In: Proceedings of the AAAI conference on artificial intelligence. vol. 36, pp. 1306–1313 (2022)
16. Li, T., Wang, L., Wu, G.: Self supervision to distillation for long-tailed visual recognition. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 630–639 (2021)

17. Lin, H., Bai, H., Liu, Z., Hou, L., Sun, M., Song, L., Wei, Y., Sun, Z.: Mope-clip: Structured pruning for efficient vision-language models with module-wise pruning error metric. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 27370–27380 (2024)
18. Lin, S., Xie, H., Wang, B., Yu, K., Chang, X., Liang, X., Wang, G.: Knowledge distillation via the target-aware transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10915–10924 (2022)
19. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. pp. 2980–2988 (2017)
20. Liu, Y., Chen, K., Liu, C., Qin, Z., Luo, Z., Wang, J.: Structured knowledge distillation for semantic segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2604–2613 (2019)
21. 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)
22. Park, W., Kim, D., Lu, Y., Cho, M.: Relational knowledge distillation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 3967–3976 (2019)
23. Peng, B., Jin, X., Liu, J., Li, D., Wu, Y., Liu, Y., Zhou, S., Zhang, Z.: Correlation congruence for knowledge distillation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5007–5016 (2019)
24. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PMLR (2021)
25. Ren, J., Yu, C., Ma, X., Zhao, H., Yi, S., et al.: Balanced meta-softmax for long-tailed visual recognition. Advances in neural information processing systems **33**, 4175–4186 (2020)
26. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: Fitnets: Hints for thin deep nets. arXiv preprint arXiv:1412.6550 (2014)
27. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision **115**, 211–252 (2015)
28. Sharma, P., Ding, N., Goodman, S., Soricut, R.: Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In: Proceedings of ACL (2018)
29. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
30. Sun, S., Ren, W., Li, J., Wang, R., Cao, X.: Logit standardization in knowledge distillation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 15731–15740 (2024)
31. Wang, X., Lian, L., Miao, Z., Liu, Z., Yu, S.X.: Long-tailed recognition by routing diverse distribution-aware experts. arXiv preprint arXiv:2010.01809 (2020)
32. Wen, X., Zhao, B., Chen, Y., Pang, J., Qi, X.: Generalization beyond data imbalance: A controlled study on clip for transferable insights. arXiv preprint arXiv:2405.21070 (2024)
33. Wu, K., Zhang, J., Peng, H., Liu, M., Xiao, B., Fu, J., Yuan, L.: Tinyvit: Fast pretraining distillation for small vision transformers. In: European conference on computer vision. pp. 68–85. Springer (2022)

34. Yang, X., Zhou, D., Feng, J., Wang, X.: Diffusion probabilistic model made slim. In: Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition. pp. 22552–22562 (2023)
35. Ye, H.J., Chen, H.Y., Zhan, D.C., Chao, W.L.: Identifying and compensating for feature deviation in imbalanced deep learning. arXiv preprint arXiv:2001.01385 (2020)
36. Zhang, C., Han, D., Qiao, Y., Kim, J.U., Bae, S.H., Lee, S., Hong, C.S.: Faster segment anything: Towards lightweight sam for mobile applications. arXiv preprint arXiv:2306.14289 (2023)
37. Zhang, Y., Xiang, T., Hospedales, T.M., Lu, H.: Deep mutual learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4320–4328 (2018)
38. Zhao, B., Cui, Q., Song, R., Liang, J.: Dot: A distillation-oriented trainer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6189–6198 (2023)
39. Zhao, B., Cui, Q., Song, R., Qiu, Y., Liang, J.: Decoupled knowledge distillation. In: Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition. pp. 11953–11962 (2022)
40. Zhou, B., Cui, Q., Wei, X.S., Chen, Z.M.: Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9719–9728 (2020)