DELVING INTO SEMANTIC SCALE IMBALANCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Model bias triggered by long-tailed data has been widely studied. However, measure based on the number of samples cannot explicate three phenomena simultaneously: **(1)** Given enough data, the classification performance gain is marginal with additional samples. **(2)** Classification performance decays precipitously as the number of training samples decreases when there is insufficient data. **(3)** Model trained on sample-balanced datasets still has different biases for different classes. In this work, we define and quantify the semantic scale of classes, which is used to measure the feature diversity of classes. It is exciting to find experimentally that there is a marginal effect of semantic scale, which perfectly describes the first two phenomena. Further, the quantitative measurement of semantic scale imbalance is proposed, which can accurately reflect model bias on multiple datasets, even on sample-balanced data, revealing a novel perspective for the study of class imbalance. Due to the prevalence of semantic scale imbalance, we propose semantic-scale-balanced learning, including a general loss improvement scheme and a dynamic re-weighting training framework that overcomes the challenge of calculating semantic scales in real-time during iterations. Comprehensive experiments show that dynamic semantic-scale-balanced learning consistently enables the model to perform superiorly on large-scale long-tailed and non-long-tailed datasets, which is a good starting point for mitigating the prevalent but unnoticed model bias.

1 INTRODUCE

In practical tasks, long-tailed class imbalance is a common problem, and the imbalance in number makes the trained model easily biased towards the dominant head classes and perform poorly on the tail classes (22; 68). However, what is overlooked is that, in addition to long-tailed data, our study finds that the model trained on sample-balanced data still shows different biases for different classes. This model bias is not taken into account by the study for class imbalance problem, and it cannot be ameliorated by the current methods proposed for long-tailed data (4; 23; 29; 60). For natural datasets, classes artificially divided by different semantic concepts correspond to different semantic scales, which can lead to different degrees of optimization when the deep metric is a single scale (46). In this study, we attempt to uncover more information from the data itself and introduce and quantify the semantic scale imbalance for representing more general model bias. The semantic-scale-balanced learning is further proposed, which is used to improve loss to mitigate model bias.

The classes corresponding to different semantic concepts have different feature diversity, and we equate the diversity to the semantic scale. Usually, the finer the semantic concept of a class label, the less rich the feature diversity, the less information a model can extract, and the worse the model performs on that class (10; 46; 6; 64; 8; 62). The manifold distribution hypothesis (49) states that a specific class of natural data is concentrated on a low-dimensional manifold. The larger the range of value variation along a specific dimension of the manifold, such as illumination and angle, the richer the feature and the larger the volume of the manifold. For example, in Figure 1, since "Swan" is a subclass of "Bird", its semantic concept is finer, so the feature diversity of "Bird" is richer than that of "Swan", and the corresponding volume of manifold is larger.

Obviously, the semantic scale can be measured by the volume of manifold. We present a reliable and numerically stable quantitative measurement of the semantic scale and further define the semantic scale imbalance. To avoid confusion, we refer to the volume of the manifold calculated on the sample space as the sample volume and the feature space as the feature volume. In addition, our

innovative study for semantic scale imbalance can simultaneously and naturally explain the following two phenomena that cannot be explicated by existing studies of class imbalance:

(1) As the number of samples increases linearly, the model performance shows a trend of rapid improvement in the early stages and leveling off in the later stages (52).

(2) Even models trained on the dataset with balanced sample numbers still suffer from class bias.

The experiments demonstrate that semantic scale imbalance is more widely present in natural datasets than sample number imbalance, allowing the study scope for class imbalance to be extended to arbitrary datasets. Then, how to mitigate the adverse effects of semantic scale imbalance? We are inspired by classification methods for long-tailed data, and current solutions for class imbalance problem usually adopt re-sampling strategies (50; 70; 5) and cost-sensitive learning (21; 63; 31). However, re-sampling may either introduce a large number of duplicate samples, making the model susceptible to overfitting when oversampling, or discard valuable samples when undersampling. Therefore by drawing on the classical re-weighting strategy (48), we propose the dynamic semantic-scale-balanced learning. Its core idea is to dynamically rather than invariably measure the degree of imbalance between semantic scales in the feature space, to achieve a dynamic evaluation of the weaker classes and assign greater weights to their corresponding losses.

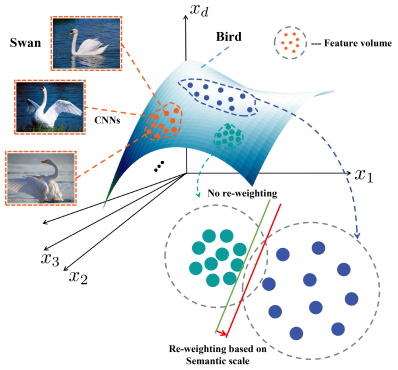


Figure 1: The features from "Bird" mapped by CNNs are concentrated on a low-dimensional manifold, and the three-color point sets represent the three sub-classes of "Bird". Among them, the orange point set represents "Swan", whose feature volume is obviously smaller than that of "Bird". The classification experiments on sample-balanced datasets show that the models are biased towards the classes with larger semantic scales, such as the decision surface shown by the green line. In this case, the re-weighting strategy based on the number of samples does NOT work, while our proposed re-weighting approach based on the semantic scale biases the decision surface toward the class with a larger feature volume (red line).

In this work, our **key contributions** are summarized as:

(1) We propose the novel idea of leveraging the volume of manifold to measure the semantic scale (Sec 3.2). It is also innovative to find that the semantic scale has the marginal effect (Sec 3.3), and that the semantic scale of the dataset is highly consistent with model performance in terms of trends.

(2) We introduce and define semantic scale imbalance, aiming to measure the degree of class imbalance by semantic scale rather than sample number, which reveals a new perspective for the study of class imbalance problem. Experiments show that semantic scale imbalance is prevalent in the dataset and can more accurately reflect model bias that affects model performance (Sec 3.4).

(3) Semantic-scale-balanced learning is proposed to mitigate model bias, which includes a general loss improvement scheme (Sec 4.1) and a dynamic re-weighting training framework (Sec 4.2) that overcomes the challenge of calculating semantic scales in real-time during iterations. Comprehensive experiments demonstrate that semantic-scale-balanced learning is applicable to a variety of datasets and achieves significant performance gains on multiple vision tasks (Sec 5).

2 Slow Drift Phenomenon of Features and Marginal Effect

Since the model parameters are changing during training, (57) studies the drifting speed of embeddings by measuring the difference in features of the same instance across training iterations. Experiments on the Stanford Online Products (SOP) dataset (36) show that the features change drastically in the early stage of training, become relatively stable after traversing the dataset twice, and drift gets extremely slowly when the learning rate decreases. This ensures that it is reasonable to leverage historical features to calculate semantic scales.

Marginal effect (10) describes that in the early stages of model training, the network is able to learn features quickly, but since there is information overlap among samples, as the number of samples

increases, the information of data will gradually saturate and the improvement in model performance from newly added samples will diminish. The effective number of samples is proposed to represent the information of data, but its limitation is that it does not work when the number of samples for each class is balanced.

Assume that the volume of each sample is unit volume 1 and define the set of all samples for a class as Ω with volume N and $N \geq 1$. A new sample may overlap with a previous sample such that the probability of overlap is P and that of non-overlap is $1-P$. As the information of data increases, the probability P will be higher.

Define the effective number of samples as E_n, and $E_n = 1 + \beta \frac{1-\beta^{n-1}}{1-\beta} = \frac{1-\beta^n}{1-\beta}$ (10), where n denotes the number of samples, hyperparameter $\beta = \frac{N-1}{N} \in [0, 1)$ controls how fast E_n grows as n increases. When $N = 1$, $\beta = 0$ and $E_n = 1$, meaning that all samples can be represented by a single prototype via data augmentation. When $N \to \infty$, $\beta \to 1$, implying that there is no overlapping, then $\lim_{\beta \to 1} \frac{1-\beta^n}{1-\beta} = \lim_{\beta \to 1} \frac{(1-\beta^n)'}{(1-\beta)'} = \lim_{\beta \to 1} \frac{-n\beta^{n-1}}{-1} = n$, which indicates that the effective number of samples does not increase faster than the number of samples, but this is not the case in our experimental results.

The effective number of samples E_n is an exponential function of the number of samples n. The hyperparameters β corresponding to classes of different grain should be different. However, the selection of β requires more information from the data itself, but this problem is not addressed by (10), which is forced to assume that β is the same for all classes. In this case, compared to the number of samples, E_n simply uses the exponential function to obtain smoother weights. Furthermore, when the number of samples is balanced, E_n is the same for each class and cannot be used to mitigate model bias, so we attempt to mine the data for information (or feature diversity) of each class to facilitate the study of imbalance problem.

3 Semantic Scale Imbalance

In this section, first, sample volume, feature volume, and semantic scale imbalance are defined. Next, we derive a quantitative measurement of feature volume to measure the semantic scale from the perspective of singular value decomposition of the data matrix and information theory. Then, the marginal effect of semantic scale is investigated. Finally, we discuss the relationship between semantic scale imbalance and model bias.

3.1 Definitions

Different semantic concepts correspond to different semantic scales, for example, the scale of "Bird" is larger than that of "Swan". For each class, we equate its feature diversity to its semantic scale and measure the semantic scale by the volume of subspace spanned by samples or features. Deep neural networks can be viewed as a combination of a feature mapping function $f(x, \theta)$ and a trained downstream classifier $g(z)$, i.e., $x \to z(\theta) \to y$. Let the samples of a class be $X = [x_1, x_2, \ldots, x_m]$, and the embeddings learned by deep neural networks are represented as $Z = \left\{ z_i | z_i = f(x_i, \theta) \in \mathbb{R}^d, i = 1, 2, \ldots, m \right\}$.

Definition 3.1. (Sample volume) The volume of the subspace spanned by sample set X.

Definition 3.2. (Feature volume) The volume of the subspace spanned by feature vectors Z.

Definition 3.3. (Semantic scale imbalance) A phenomenon of imbalance in the size of semantic scales measured by sample volume or feature volume.

3.2 Quantification of Semantic Scale

Given the data $X = [x_1, x_2, \ldots, x_m]$ and the learned embeddings $Z = [z_1, z_2, \ldots, z_m] \in \mathbb{R}^{d \times m}, z_i = f(x_i, \theta) \in \mathbb{R}^d, i = 1, 2, \ldots, m$, the volume of subspace spanned by the random vector z_i (i.e., the feature volume) is derived below, and the sample volume can be calculated in the same way (Appendix E). The covariance matrix of random vector z_i is estimated as $\Sigma = E\left[\frac{1}{m} \sum_{i=1}^{m} z_i z_i^T\right] = \frac{1}{m} Z Z^T \in \mathbb{R}^{d \times d}$, $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d > 0$ are the eigenvalues

of real symmetric matrix. The singular value decomposition (SVD) of Z yields $Z = U\Sigma V^T$ and the singular values are $\sigma_j = \sqrt{\lambda_j}, j = 1, 2, \ldots, d$. The volume of the space spanned by the vector z_i is proportional to the product of all singular values of the feature matrix Z, i.e., $Vol(Z) \propto \prod_{j=1}^{d} \sigma_j = \sqrt{\prod_{j=1}^{d} \lambda_j}$. After the determinant expansion, the characteristic polynomial of the matrix Σ is $\Phi(\lambda) = \det(\lambda I - \Sigma) = \lambda^d - (a_{11} + a_{22} + \cdots a_{dd})\lambda^{d-1} + \cdots + (-1)^d \det\Sigma$, and $\lambda_1\lambda_2 \cdots \lambda_d = \det\Sigma$. Therefore, the volume of the space spanned by the vector z_i is proportional to the square root of the determinant of the covariance matrix of Z:

$$Vol(Z) \propto \sqrt{\det(\frac{1}{m}ZZ^T)}. \tag{1}$$

The same result can be derived from the volume of a parallel hexahedron defined by vectors (Appendix G). Considering that real-world metric tools typically have a dynamic range, for example, a ruler always has multiple scales (1mm, 1cm, or even 10cm) to measure objects of different scales. We expect the quantitative measurement of feature volume to have a multi-scale metric and therefore use the sphere packing method (7; 32), which is normally adopted in information theory, to implement it.

The feature volume is positively correlated with the number of spheres, and since the sphere packing has error at the subspace boundary, the error of the finite feature vectors is assumed to be independent additive Gaussian noise: ${z_i}' = z_i + w_i$, where $w_i \sim N(0, \frac{\varepsilon^2}{n}I)$. The estimate of the number of spheres of radius ε needed to pack the space spanned by all vectors is N_ε=$Vol\,(Z')/Vol\,(w)$. The adjustment of the metric scale can be achieved by tuning the radius ε of the spheres, thus controlling the measurement result of feature volume. Then the covariance matrix of the vector z_i is $\Sigma' = \frac{\varepsilon^2}{n}I + \frac{1}{m}ZZ^T \in \mathbb{R}^{d\times d}$, such that $Vol(Z') \propto \sqrt{\det(\frac{\varepsilon^2}{n}I + \frac{1}{m}ZZ^T)}$ and $Vol(w) \propto \sqrt{\det(\frac{\varepsilon^2}{n}I)}$. The feature volume is proportional to N_ε:

$$\begin{aligned} Vol\,(Z) \propto N_\varepsilon = \frac{Vol\,(Z')}{Vol\,(w)} &= \sqrt{\frac{\det\left(\frac{\varepsilon^2}{n}I + \frac{1}{m}ZZ^T\right)}{\det\left(\frac{\varepsilon^2}{n}I\right)}} \\ &= \sqrt{\det\left(I + \frac{n}{m\varepsilon^2}ZZ^T\right)}. \end{aligned} \tag{2}$$

The dimension of feature z_i is d, so let $n = d$. In order to increase the numerical stability, we perform a logarithmic transformation of the above equation, which does not affect the monotonicity of the function, and we can obtain $Vol\,(Z) \propto \log_2\sqrt{\det\left(I + \frac{n}{m\varepsilon^2}ZZ^T\right)} = \frac{1}{2}\log\det\left(I + \frac{d}{m\varepsilon^2}ZZ^T\right)$. In practical training, it is essential to normalize the feature vectors so that their mean value is 0. In this work, we set $\varepsilon = 1000$, and the value of ε does not affect the relative size of the space spanned by feature vectors of each class. The volume of the space spanned by Z can be can be written as:

$$Vol(Z) = \frac{1}{2}\log_2\det(I + \frac{d}{m}(Z - Z_{mean})(Z - Z_{mean})^T), \tag{3}$$

where Z_{mean} is the mean value of Z, $Vol(Z) > 0$ when the number of samples $m > 1$. We measure the semantic scale S' by the feature volume, i.e., $S' = Vol(Z)$, and the larger S', the richer the feature diversity, which we verify in Appendix C using multiple Stanford point cloud manifolds.

3.3 Marginal Effect of Semantic Scale

The marginal effect describes that the feature richness will gradually saturate as the number of samples increases, so the change of semantic scale should also conform to the marginal effect. Figure 2 illustrates that as the number of samples increases, the semantic scale S' measured by sample volume gradually saturates, which indicates that the quantitative measurement of semantic scale is reasonable. In addition, the growth rate of the semantic scale varies across classes, which is determined by the grain size of the class itself. It leads to different semantic scales even if all classes have the same number of samples.

To investigate why adding samples has a marginal improvement in model performance when the training samples are sufficient, we use the following method to generate new training datasets for

classification experiments: assume that the total number of classes in the original dataset is C, and m samples are randomly selected from each class to form a sub-dataset with a total number of samples of $C \times m$. The sub-datasets generated based on CIFAR-10, CIFAR-100 (26) and Mini-ImageNet (53) are shown in Appendix D.1.

We train ResNet-18 and ResNet-34 (14) on each of the **31** training sets in Table 6, and the sum of the semantic scales for all classes and the corresponding top-1 accuracy are shown in Figure 2. We are pleasantly surprised to find that when the semantic scale increases rapidly, the model performance improves swiftly with it, and when the semantic scale becomes saturated, the improvement is small.

3.4 Semantic Scale Imbalance and Model Bias

3.4.1 Quantification of Semantic Scale Imbalance

The previous subsection shows that the sum of semantic scales for all classes in the dataset is highly correlated with the model performance, and we further investigate the relationship between semantic scale and model bias for different classes. When a class is closer to other classes, the model performs worse on that class (35; 1). Therefore, inter-class interference is additionally considered when quantifying the degree of imbalance between semantic scales of classes. When class i is closer to other classes, a smaller weight w_i is applied to the semantic scale of class i.

Specifically, the semantic scale of m classes after maximum normalization is assumed to be $S' = [S'_1, S'_2, \ldots, S'_m]^T$, and the centers of all classes are $C = [c_1, c_2, \ldots, c_m]^T$. Define the distance between the centers of class i and class j as $d_{i,j} = \|c_i - c_j\|_2$, the weight $w_i = \frac{1}{m-1}\sum_{j=1}^{m}\|c_i - c_j\|_2$. The weights of m classes are written as $W' = [w_1, w_2, \ldots, w_m]^T$. After the maximum normalization and logarithmic transformation of W', we can obtain $W = \log_2(\alpha + W'), \alpha \geq 1$, where α is used to control the smoothing degree of W. After considering the inter-class distance, the semantic scale $S = S' \odot W$, and the role of S' in dominating the degree of imbalance is greater when α is larger.

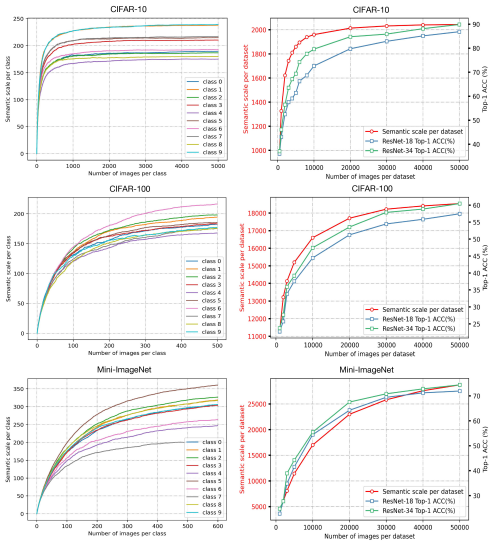


Figure 2: **Left column**: curves of semantic scales with increasing number of samples for the first ten classes from different datasets. **Right column**: for different sub-datasets, curves of the sum of semantic scales for all classes and top-1 accuracy curves of trained ResNet-18 and ResNet-34. All models are trained using the Adam optimizer (24) with an initial learning rate of 0.01 and then decayed by 0.98 at each epoch.

To obtain the most appropriate α, we calculate the Pearson correlation coefficients between the semantic scale and the accuracy of ResNet-18 and ResNet-34 trained on CIFAR-10-LT and CIFAR-10, as shown in Table 1. The experimental settings are in Appendix D.2. It can be found that S' is more dominant on long-tailed data than on non-long-tailed data, and the improved S is better than S' and far better than the number of samples in reflecting model bias. In the following experiments, we let α be 2 on the long-tailed data and 1 on the non-long-tailed data.

Table 1: Pearson correlation coefficients between the accuracy of classes and the semantic scales S with different α. N denotes the number of samples, and S' represents the semantic scale without considering inter-class interference.

Dataset	Model	N	W	S'	S				
					α=1	α=1.5	α=2	α=2.5	α=3
CIFAR-10-LT	ResNet-18	0.8346	0.2957	0.8688	0.8456	**0.9603**	0.9553	0.9398	0.9269
	ResNet-34	0.7938	0.3186	0.9426	0.7950	0.9678	**0.9884**	0.9854	0.9796
CIFAR-10	ResNet-18	0.0950	0.1743	0.5433	**0.7850**	0.7250	0.6644	0.6060	0.5607
	ResNet-34	0.1502	0.2075	0.5750	**0.8056**	0.7442	0.6870	0.6465	0.5906

3.4.2 SEMANTIC SCALE IMBALANCE ON LONG-TAILED DATA

Previous studies have roughly attributed model bias to the imbalance in the number of samples. The experimental results in the first row of Figure 3 show that even though the number of MNIST-LT-1 is similar to that of MNIST-LT-2, the class-wise accuracy on MNIST-LT-1 is closer to that on MNIST, just as their semantic scales are also more similar.

In addition, Figure 3 indicate that models on certain classes with fewer samples outperform those on classes with more samples, and that the semantic scale S reflects model bias more accurately. Further, we also observe that the accuracy of the CIFAR-100-LT does not show a significant decreasing trend, which can be explained by the marginal effect of the semantic scale (Sec 3.3).

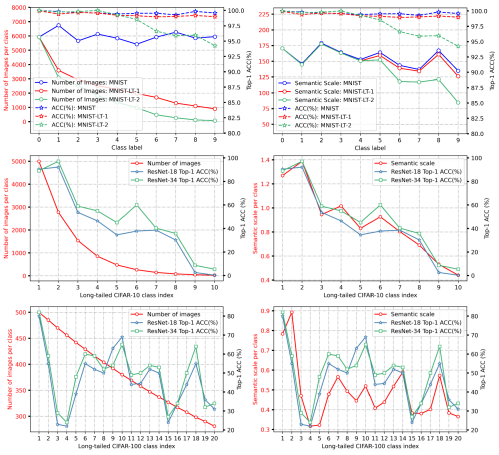


Figure 3: Correlation study of accuracy with the number of samples and semantic scale on MNIST, MNIST-LT (Appendix D.3), CIFAR-10-LT and CIFAR-100-LT datasets.

3.4.3 SEMANTIC SCALE IMBALANCE ON NON-LONG-TAILED DATA

Figure 4 demonstrates the model bias not only on long-tailed data but also on sample-balanced data. Usually, the classes with smaller semantic scales have lower accuracies. Depending on the size of the semantic scale, it can make it possible for the weaker and dominant classes to be well differentiated. It should be noted that the weaker classes are not random, and experiments in Figure 4 show that the models always perform worse on the same classes. More semantic scale imbalance of the datasets is shown in Appendix D.4.

In summary, semantic scale imbalance can represent model bias more generally and appropriately, and further, we expect to improve the overall performance of the model when facing the semantic scale imbalance problem. Therefore, we propose the dynamic semantic-scale-balanced learning by drawing on the re-weighting strategy.

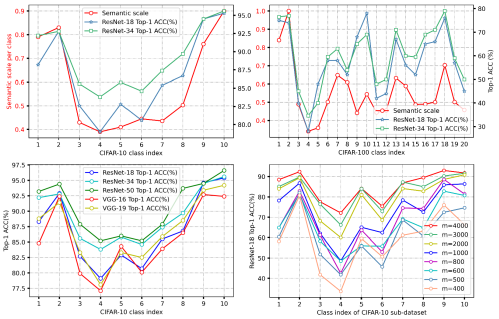


Figure 4: **Top row**: correlation study between accuracy and semantic scale on the sample-balanced dataset. **Bottom row**: performance of different models (42) trained on the CIFAR-10 dataset and performance of ResNet-18 trained on different sub-datasets of CIFAR-10 from Table 6 of Appendix D.1.

4 DYNAMIC SEMANTIC-SCALE-BALANCED LEARNING

Deep neural networks can be viewed as a combination of the feature mapping function and the classifier, and several studies have shown that model bias is mainly caused by classifier bias (70; 58; 2), so we are more concerned with semantic scale imbalance in the feature space, i.e., semantic scale measured by the **feature volume**. In this section, we propose a general semantic-scale-based loss improvement scheme and design a training framework for the successful application of the scheme.

4.1 DYNAMIC SEMANTIC-SCALE-BALANCED LOSS

During training, the feature vectors corresponding to the samples vary with the model parameters, and thus the semantic scale per class is constantly changing. Compared with the traditional re-weighting strategy, we propose to calculate the degree of imbalance between semantic scales in real-time at each iteration in order to dynamically evaluate the weaker classes and assign greater weights to their corresponding losses. Specifically, for class i at each iteration, normalized re-weighting terms $\alpha_i \propto \frac{1}{S_i}, \sum_{i=1}^{C} \alpha_i = 1$, inversely proportional to the semantic scales that take into account

inter-class interference, are introduced, and C is the total number of classes. Given the embedding z of a sample and label y_i, the dynamic semantic-scale-balanced (**DSB**) loss can be expressed as $DSB(z, y_i) = \frac{1}{S_i} L(z, y_i), i = 1, 2, \ldots, C$, where y_i is the label of the sample from class i. How to combine general loss to generate DSB loss is described in Appendix F.1. **It should be noted** that all methods that re-balance the loss function or adjust the sampling rate based on the sample number can be improved with the semantic scale, since both the semantic scale and the sample number are natural measures and they are not model-dependent.

However, the number of samples used at each iteration is limited, and it is not possible to obtain the features of all samples for calculating the semantic scales. Therefore, we propose a dynamic re-weighting training framework that enables DSB loss to be successfully applied.

4.2 Dynamic Re-Weighting Training Framework

Inspired by the slow drift phenomenon of features (57; 30), we design a storage pool Q to store and update historical features and propose a three-stage training framework. A mini-batch of features can be dynamically updated at each iteration, and the semantic scale of each class is calculated using all the features in the storage pool. The three-stage training framework is shown in Figure 9 and Algorithm 2 (More details are in Appendix F.2), with the following textual description.

(1) In the **first stage**, all the features and labels generated by the 1st epoch are stored in Q, but they cannot be used directly to calculate semantic scales due to the large drift of historical features from current features in the early stage.

(2) The **second stage** corresponds to epoch 2 to epoch n. At each iteration, the oldest mini-batch features and labels in Q are removed and those generated by the current iteration are stored. The goal is to continuously update the features in Q until the feature drift is small enough. We set n to 5 in our experiments, and the original loss function is used in the first two stages. Figure 5 shows the effect of n on the model performance. A larger n does not hurt the model performance, but only takes a little more time. Experience suggests that setting n to 5 is sufficient.

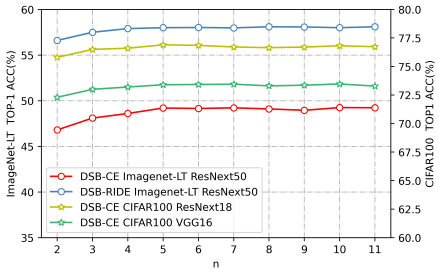


Figure 5: The performance of different models with different losses on different datasets for different values of n.

(3) The **third stage** corresponds to epoch $> n$. At each iteration, the semantic scales are calculated using the features in Q after updating Q, and the original loss is re-weighted.

The comparison and analysis of the **video memory** and **training speed** are in Appendix F.2. We answer possible questions about the methods section in detail in Appendix B.

5 Experiments

To validate the superiority and generality of the proposed dynamic semantic-scale-balanced learning, we design three experiments. The **first experiment** is conducted on large-scale long-tailed datasets, ImageNet-LT and iNaturalist2018 (51), to confirm the superior performance of our approach on long-tailed data. The **second experiment** uses large-scale ImageNet (40) and sample-balanced CIFAR-100, and the **third experiment** selects CIFAR-100-LT and benchmark datasets commonly used in deep metric learning (CUB-2011 (54) and Cars196 (25)). The **latter two experiments** demonstrate the generality of our approach on non-long-tailed data. More experimental results are provided in Appendix D.5.

5.1 Results on ImageNet-LT and iNaturalist2018

ImageNet-LT is a long-tailed version of ImageNet containing 1,000 classes with between 1,280 and 5 samples per class. iNaturalist2018 is a real-world, extremely unbalanced dataset containing 437,513 images from 8,142 classes. For fair comparisons, we adopt the official training and validation splits.

Table 2 shows that when CE, Focal (28) and RIDE are combined with our approach (Appendix D.1), the model overall performance is significantly improved. For example, the overall accuracy of

Table 2: Top-1 Acc(%) on ImageNet-LT and iNaturalist2018. We use ResNext-50 (61) on ImageNet-LT and ResNet-50 (14) on iNaturalist2018 as the network backbone for all methods. And we conduct model training with the SGD optimizer based on batch size 256 (for ImageNet-LT) / 512 (for iNaturalist), momentum 0.9, weight decay factor 0.0005, and learning rate 0.1 (linear LR decay).

Methods	ImageNet-LT(ResNeXt50)				iNaturalist 2018(ResNet50)			
	Head	Middle	Tail	Overall	Head	Middle	Tail	Overall
BBN (70)	43.3	45.9	43.7	44.7	49.4	70.8	65.3	66.3
PaCo (9)	63.2	51.6	39.2	54.4	69.5	72.3	73.1	72.3
DIVE (15)	64.1	50.4	31.5	53.1	70.6	70	67.6	69.1
CE	65.9	37.5	7.70	44.4	67.2	63.0	56.2	61.7
CB-CE (10)	39.6	32.7	16.8	33.2	53.4	54.8	53.2	54.0
DSB-CE	**67.3**	**42.5**	**21.4**(+13.7)	**49.2**(+4.8)	**68.5**	**63.4**	**62.7**(+6.5)	**64.3**(+2.6)
Focal (10)	67.0	41.0	13.1	47.2	-	-	-	61.1
CB-Focal (10)	-	-	-	-	-	-	-	61.2
DSB-Focal	**68.1**	**44.2**	**23.7**(+10.6)	**50.6**(+3.4)	**70.6**	**62.8**	**58.4**	**63.5**(+2.4)
LDAM (3)	60.0	49.2	31.9	51.1	-	-	-	64.6
DSB-LDAM	**60.7**	**50.5**	**33.4**(+1.5)	**52.3**(+1.2)	**69.4**	**66.5**	**61.9**	**65.7**(+1.1)
BS (38)	47.8	37.7	24.9	40.4	60.1	51.4	46.7	53.2
DSB-BS	**48.5**	**39.3**	**28.7**(+3.8)	**42.9**(+2.5)	**61.4**	**52.8**	**49.4**(+2.7)	**55.1**(+1.9)
LADE (16)	62.3	49.3	31.2	51.9	-	-	-	69.7
DSB-LADE	**62.6**	**50.4**	**33.6**(+2.4)	**53.2**(+1.3)	**72.3**	**70.7**	**65.8**	**70.5**(+0.8)
MBJ (30)	61.6	48.4	39.0	52.1	-	-	-	70.0
DSB+MBJ	**63.2**	**49.6**	**40.7**(+1.7)	**53.3**(+1.2)	**73.6**	**70.2**	**66.2**	**70.9**(+0.9)
RIDE (56)	67.9	52.3	36.0	56.1	70.9	72.4	73.1	72.6
MBJ+RIDE (30)	68.4	54.1	37.7	57.7	-	-	-	73.0
DSB+RIDE	**68.6**	**54.5**	**38.5**(+2.5)	**58.2**(+2.1)	**70.7**	**74.0**	**74.2**(+1.1)	**73.4**(+0.8)

DSB-CE is **4.8%** and **2.6%** higher than CE on ImageNet-LT and iNaturalist2018, respectively. We also report the performance on three subsets (Head: more than 100 images, Middle: 20-100 images, Tail: less than 20 images) of these two datasets. It can be observed that our proposed method has the largest improvement for the tail subset without compromising the performance of the head subset, where DSB-CE and DSB-Focal improve **13.7%** and **10.6%**, respectively, over the original method in the tail subset of ImageNet-LT, effectively alleviating the model bias.

5.2 Results on ImageNet and CIFAR-100

Table 3: Comparison on ImageNet and CIFAR-100. On ImageNet, we use random clipping, mixup (67), and cutmix (66) to augment the training data, and all models are optimized by Adam with batch size of 512, learning rate of 0.05, momentum of 0.9, and weight decay factor of 0.0005. On CIFAR-100, we set the batch size to 64 and augment the training data using random clipping, mixup, and cutmix. An Adam optimizer with learning rate of 0.1 (linear decay), momentum of 0.9, and weight decay factor of 0.005 is used to train all networks.

	ImageNet Top-1 Acc(%)			CIFAR-100 Top-1 Acc(%)		
Methods	CE	DSB-CE	Δ	CE	DSB-CE	Δ
VGG16 (42)	71.6	72.9	**+1.3**	71.9	73.4	**+1.5**
BN-Inception (47)	73.5	74.4	**+0.9**	74.1	75.2	**+1.1**
ResNet-18	70.1	71.2	**+1.1**	75.6	76.9	**+1.3**
ResNet-34	73.5	74.3	**+0.8**	76.8	77.9	**+1.1**
ResNet-50	76.0	76.8	**+0.8**	77.4	78.3	**+0.9**
DenseNet-201 (19)	77.2	78.1	**+0.9**	78.5	79.7	**+1.2**
SE-ResNet-50 (18)	77.6	78.4	**+0.8**	78.6	79.3	**+0.7**
ResNeXt-101 (61)	78.8	79.7	**+0.9**	77.8	78.8	**+1.0**

We use the ILSVRC2012 split contains 1,281,167 training and 50,000 validation images. Each class of CIFAR-100 contains 500 images for training and 100 images for testing. The results in Table 3 indicate that our approach is able to achieve performance gains greater than **1%** for a variety of networks on both datasets. In particular, it enables VGG16 to improve **1.3%** and **1.5%** on ImageNet and CIFAR-100, respectively, compared to the original method. This implies that there is a semantic scale imbalance in non-long-tailed datasets and it affects the model performance.

5.3 RESULTS ON CUB-2011, CARS196 AND CIFAR-100-LT

Since we also improve on the classical losses (NormSoftmax and SoftTriple (37)) in the field of deep metric learning, we abide by the widely adopted backbone network, experimental parameters, and the division of datasets in this field (Appendix D.5). The two improved loss functions are denoted as DSB-NSM and DSB-ST, respectively, and their formulas are given in Appendix F.1.

Table 4: Results on CUB-2011 and Cars196. We evaluate the model performance with Recall@K (36) and Normalized Mutual Information (NMI) (41).

Dataset		CUB-2011			Cars196		
Metric		R@1	R@2	NMI	R@1	R@2	NMI
dim 64	NormSoftmax	57.8	70.0	65.3	76.8	85.6	66.7
	DSB-NSM	**59.2**(+1.4)	**70.7**(+0.7)	**66.5** (+1.2)	**77.9**(+1.1)	**86.4**(+0.8)	**67.8**(+1.1)
	SoftTriple	60.1	71.9	66.2	78.6	86.6	67.0
	DSB-ST	**61.3** (+1.2)	**72.7**(+0.8)	**67.3**(+1.1)	**79.8**(+1.2)	**87.5**(+0.9)	**68.3** (+1.3)
dim 512	Circle	66.7	77.4	-	83.4	89.8	-
	NormSoftmax	63.9	75.5	68.3	83.2	89.5	69.7
	DSB-NSM	**65.1**(+1.2)	**76.3**(+0.8)	**69.2**(+0.9)	**84.0**(+0.8)	**90.2**(+0.7)	**70.9**(+1.2)
	SoftTriple	65.4	76.4	69.3	84.5	90.7	70.1
	DSB-ST	**66.4**(+1.0)	**77.0**(+0.6)	**70.6**(+1.3)	**85.6**(+1.1)	**91.3**(+0.6)	**71.1**(+1.0)

Results on CUB-2011 and Cars196. Table 4 summarizes the performance of our method with 64 and 512 embeddings, respectively. The experiments show that DSB loss is able to consistently improve by more than **1%** on R1 and NMI. DSB-ST with 512 embeddings performs superiorly on Cars196, where R@1 and R@2 exceed the Circle loss (45) by **2.2%** and **1.5%**, respectively.

Table 5: Results on CIFAR-100-LT. The imbalance factor of a dataset is defined as the value of the number of training samples in the largest class divided by that in the smallest class.

Dataset		CIFAR-100-LT								
Imbalance factor		10			50			200		
Metric		R@1	R@2	NMI	R@1	R@2	NMI	R@1	R@2	NMI
dim 64	NormSoftmax	54.6	65.2	62.4	49.6	60.5	58.0	43.4	54.5	52.9
	CB-NSM	55.7	66.1	63.3	50.5	61.1	58.7	45.5	55.3	53.8
	DSB-NSM	**56.3**	**66.7**	**63.5**	**51.3**	**61.4**	**59.1**	**46.0**	**56.1**	**54.4**
	SoftTriple	56.6	67.6	63.9	49.5	61.0	58.3	46.6	57.8	55.4
	CB-ST	58.1	68.4	65.1	51.1	62.8	59.4	48.2	59.5	56.6
	DSB-ST	**58.8**	**69.0**	**65.8**	**51.5**	**62.5**	**59.7**	**49.3**	**60.6**	**57.3**

Results on CIFAR-100-LT. Class-balanced loss (CB loss) that performs well on long-tailed data and is also based on the re-weighting strategy is selected for comparison with DSB loss. Analyzing the results in Table 5, the DSB loss outperforms the CB loss overall. Among them, when the imbalance factor of long-tailed CIFAR-100 is 200, DSB-ST performs significantly better than CB-ST, with higher performance than SoftTriple on R@1, R@2 and NMI by **2.7%**,**2.8%** and **1.9%**.

Experiment Summary. Extensive experiments confirm that dynamic semantic-scale-balanced learning has superior performance not only on long-tailed datasets, but also on non-long-tailed datasets, and even on sample-balanced datasets. This also means that the semantic scale imbalance needs to be paid extensive attention.

6 DISCUSSION

In this work, we pioneer the concept and quantitative measurement of semantic scale imbalance, and make two important discoveries: **(1)** semantic scale has marginal effects, and **(2)** semantic scale imbalance can accurately describe model bias. It is important to note that **our proposed semantic scale, like the number of samples, is a natural measure of class imbalance and does not depend on the model's predictions** (See Related Work in Appendix A). Semantic scale can guide data augmentation, e.g., semantic scale imbalance can evaluate which classes are the weaker classes that need to be augmented, and marginal effects can assist us to select a more appropriate number of samples. We expect that our work will bring more attention to the more prevalent model bias, improve the robustness of models and promote the development of fairer AI.

REFERENCES

[1] Elie Aljalbout, Vladimir Golkov, Yawar Siddiqui, Maximilian Strobel, and Daniel Cremers. Clustering with deep learning: Taxonomy and new methods. *arXiv preprint arXiv:1801.07648*, 2018.

[2] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in neural information processing systems*, 32, 2019.

[3] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in neural information processing systems*, 32, 2019.

[4] 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.

[5] Hsin-Ping Chou, Shih-Chieh Chang, Jia-Yu Pan, Wei Wei, and Da-Cheng Juan. Remix: Rebalanced mixup. In *European Conference on Computer Vision*, pages 95–110. Springer, 2020.

[6] Peng Chu, Xiao Bian, Shaopeng Liu, and Haibin Ling. Feature space augmentation for long-tailed data. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16*, pages 694–710. Springer, 2020.

[7] Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.

[8] Ekin Dogus Cubuk, Ethan S Dyer, Rapha Gontijo Lopes, and Sylvia Smullin. Tradeoffs in data augmentation: An empirical study. 2021.

[9] Jiequan Cui, Zhisheng Zhong, Shu Liu, Bei Yu, and Jiaya Jia. Parametric contrastive learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 715–724, 2021.

[10] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9268–9277, 2019.

[11] Charles Elkan. The foundations of cost-sensitive learning. In *International joint conference on artificial intelligence*, volume 17, pages 973–978. Lawrence Erlbaum Associates Ltd, 2001.

[12] Andrew Estabrooks, Taeho Jo, and Nathalie Japkowicz. A multiple resampling method for learning from imbalanced data sets. *Computational intelligence*, 20(1):18–36, 2004.

[13] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing*, pages 878–887. Springer, 2005.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[15] Yin-Yin He, Jianxin Wu, and Xiu-Shen Wei. Distilling virtual examples for long-tailed recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 235–244, 2021.

[16] Youngkyu Hong, Seungju Han, Kwanghee Choi, Seokjun Seo, Beomsu Kim, and Buru Chang. Disentangling label distribution for long-tailed visual recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6626–6636, 2021.

[17] Ting-I Hsieh, Esther Robb, Hwann-Tzong Chen, and Jia-Bin Huang. Droploss for long-tail instance segmentation. In *AAAI*, volume 3, page 15, 2021.

[18] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[19] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[20] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[21] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7610–7619, 2020.

[22] Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002.

[23] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. *arXiv preprint arXiv:1910.09217*, 2019.

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[25] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013.

[26] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[27] Xiaotong Li, Yongxing Dai, Yixiao Ge, Jun Liu, Ying Shan, and Ling-Yu Duan. Uncertainty modeling for out-of-distribution generalization. *arXiv preprint arXiv:2202.03958*, 2022.

[28] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[29] Jialun Liu, Yifan Sun, Chuchu Han, Zhaopeng Dou, and Wenhui Li. Deep representation learning on long-tailed data: A learnable embedding augmentation perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2970–2979, 2020.

[30] Jialun Liu, Jingwei Zhang, Wenhui Li, Chi Zhang, Yifan Sun, et al. Memory-based jitter: Improving visual recognition on long-tailed data with diversity in memory. *arXiv preprint arXiv:2008.09809*, 2020.

[31] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2537–2546, 2019.

[32] Yi Ma, Harm Derksen, Wei Hong, and John Wright. Segmentation of multivariate mixed data via lossy data coding and compression. *IEEE transactions on pattern analysis and machine intelligence*, 29(9):1546–1562, 2007.

[33] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European conference on computer vision (ECCV)*, pages 181–196, 2018.

[34] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[35] Erxue Min, Xifeng Guo, Qiang Liu, Gen Zhang, Jianjing Cui, and Jun Long. A survey of clustering with deep learning: From the perspective of network architecture. *IEEE Access*, 6:39501–39514, 2018.

[36] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4004–4012, 2016.

[37] Qi Qian, Lei Shang, Baigui Sun, Juhua Hu, Hao Li, and Rong Jin. Softtriple loss: Deep metric learning without triplet sampling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6450–6458, 2019.

[38] Jiawei Ren, Cunjun Yu, Xiao Ma, Haiyu Zhao, Shuai Yi, et al. Balanced meta-softmax for long-tailed visual recognition. *Advances in neural information processing systems*, 33:4175–4186, 2020.

[39] Jiawei Ren, Cunjun Yu, Xiao Ma, Haiyu Zhao, Shuai Yi, et al. Balanced meta-softmax for long-tailed visual recognition. *Advances in Neural Information Processing Systems*, 33:4175–4186, 2020.

[40] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[41] Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge, 2008.

[42] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[43] Adarsh Subbaswamy, Roy Adams, and Suchi Saria. Evaluating model robustness and stability to dataset shift. In *International Conference on Artificial Intelligence and Statistics*, pages 2611–2619. PMLR, 2021.

[44] Yanmin Sun, Mohamed S Kamel, Andrew KC Wong, and Yang Wang. Cost-sensitive boosting for classification of imbalanced data. *Pattern recognition*, 40(12):3358–3378, 2007.

[45] Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei. Circle loss: A unified perspective of pair similarity optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6398–6407, 2020.

[46] Yifan Sun, Yuke Zhu, Yuhan Zhang, Pengkun Zheng, Xi Qiu, Chi Zhang, and Yichen Wei. Dynamic metric learning: Towards a scalable metric space to accommodate multiple semantic scales. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5393–5402, 2021.

[47] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[48] Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. Equalization loss for long-tailed object recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11662–11671, 2020.

[49] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

[50] Ivan Tomek et al. Two modifications of cnn. 1976.

[51] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8769–8778, 2018.

[52] Grant Van Horn and Pietro Perona. The devil is in the tails: Fine-grained classification in the wild. *arXiv preprint arXiv:1709.01450*, 2017.

[53] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29:3630–3638, 2016.

[54] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[55] Tong Wang, Yousong Zhu, Chaoyang Zhao, Wei Zeng, Jinqiao Wang, and Ming Tang. Adaptive class suppression loss for long-tail object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3103–3112, 2021.

[56] Xudong Wang, Long Lian, Zhongqi Miao, Ziwei Liu, and Stella X Yu. Long-tailed recognition by routing diverse distribution-aware experts. *arXiv preprint arXiv:2010.01809*, 2020.

[57] Xun Wang, Haozhi Zhang, Weilin Huang, and Matthew R Scott. Cross-batch memory for embedding learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6388–6397, 2020.

[58] Yiru Wang, Weihao Gan, Jie Yang, Wei Wu, and Junjie Yan. Dynamic curriculum learning for imbalanced data classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5017–5026, 2019.

[59] Tong Wu, Qingqiu Huang, Ziwei Liu, Yu Wang, and Dahua Lin. Distribution-balanced loss for multi-label classification in long-tailed datasets. In *European Conference on Computer Vision*, pages 162–178. Springer, 2020.

[60] Liuyu Xiang, Guiguang Ding, and Jungong Han. Learning from multiple experts: Self-paced knowledge distillation for long-tailed classification. In *European Conference on Computer Vision*, pages 247–263. Springer, 2020.

[61] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[62] Shuo Yang, Lu Liu, and Min Xu. Free lunch for few-shot learning: Distribution calibration. *arXiv preprint arXiv:2101.06395*, 2021.

[63] Yuzhe Yang and Zhi Xu. Rethinking the value of labels for improving class-imbalanced learning. *arXiv preprint arXiv:2006.07529*, 2020.

[64] Xi Yin, Xiang Yu, Kihyuk Sohn, Xiaoming Liu, and Manmohan Chandraker. Feature transfer learning for face recognition with under-represented data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5704–5713, 2019.

[65] Yaodong Yu, Kwan Ho Ryan Chan, Chong You, Chaobing Song, and Yi Ma. Learning diverse and discriminative representations via the principle of maximal coding rate reduction, 2020.

[66] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019.

[67] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[68] Yifan Zhang, Bingyi Kang, Bryan Hooi, Shuicheng Yan, and Jiashi Feng. Deep long-tailed learning: A survey. *arXiv preprint arXiv:2110.04596*, 2021.

[69] Zizhao Zhang and Tomas Pfister. Learning fast sample re-weighting without reward data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 725–734, 2021.

[70] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. 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*, pages 9719–9728, 2020.

[71] Zhi-Hua Zhou and Xu-Ying Liu. Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on knowledge and data engineering*, 18(1):63–77, 2005.

A RELATED WORK

Real-world datasets tend to long-tailed. The extreme imbalance in the number of samples for long-tailed data prevents the classification model from learning the distribution of the tail classes adequately, which leads to poor performance on the tail classes. Therefore, the methods of re-balancing the number of samples (23; 13; 12; 69) and balancing the loss incurred per class (11; 71; 44), i.e., re-sampling and cost-sensitive learning, are proposed. Among them cost-sensitive learning is most relevant to our work.

(39) proposes to use the frequency of labels to adjust the loss during training to mitigate class bias. (28) assigns weights to the loss for each class, and the hard samples are given higher weights. Recent studies have shown that re-weighting the loss strictly by the inverse of the number of samples is moderate (33; 34). Some methods that generate weights for re-weighting in a more "smooth" manner perform better, such as taking the square root of the number of samples (33) as weights. CB loss (10) attributes the better performance of this more "smooth" approach to the presence of marginal effects, while other studies (48; 17; 55) attribute it to the negative gradient over-suppression. Distribution-balanced loss (59) proposes negative-tolerant regularization to mitigate gradient differences, and recalculates the loss by calculating the ratio of the expected to the actual sampling frequency for each class. Seesaw Loss (55) leverages mitigation and compensation factors to dynamically suppress excessive negative sample gradients on the tail classes while complementing the penalty for misclassified samples. Furthermore, with the proposal of decoupled training (70), (23) adopts decoupled training, using cross-entropy loss to learn features in the first stage and re-balancing loss to learn the classifier in the second stage.

Unlike the above studies of re-balancing loss, which all re-balance loss based on the number of samples or the ratio of positive and negative gradients, our work proposes a novel measure, called semantic scale. Compared to the number of samples, the semantic scale also considers the sample distribution scope. In contrast to gradient-based measures, the semantic scale does not depend on the model output and gradient back-propagation, and is a natural measure similar to the number of samples. Work on model robustness has focused on the out-of-domain generalization performance of models, an area known as "out-of-distribution generalization of models". For example, (43) aims to maintain the good performance of the model when the test distribution deviates from the training distribution. Similarly, (27) aims to allow the model to learn more information outside the domain. Unlike them, we are concerned with the problem that model bias introduced by unbalanced data makes the model perform poorly in certain classes.

B EXPLANATION OF A FEW KEY POINTS

B.1 HOW DOES THE SECTION "SLOW DRIFT PHENOMENON AND MARGINAL EFFECTS OF CHARACTERISTICS" RELATE TO THE REST OF THE PAPER?

(1) Association with effective number of samples

The effective number of samples discusses the relationship between the sample number and feature diversity in a class, and it argues that feature diversity has marginal effects. However, the effective number of samples has many major drawbacks (introduced in Section 2), such as it does not work on sample-balanced datasets and does not give a quantitative measure of feature diversity. We therefore propose the semantic scale to measure feature diversity, which perfectly fits the marginal effects, and is the first quantitative measure of feature diversity. In addition, our approach simultaneously explains three phenomena that cannot be explicated by other methods, which indicates the reliability of our proposed method.

(2) Association with feature slow drift

Since experiments show a very high correlation between semantic scale and model bias, we propose to re-weight the loss function with the inverse of the semantic scale. The features are dynamically changing during training, and all the feature vectors are needed to calculate the semantic scale of each class. Obviously, the data in one batch is not enough, so we propose to dynamically update and store the historical features to calculate the semantic scale, and the feature slow drift phenomenon ensures the feasibility of this operation.

Section 2 is indispensable for the whole paper, it ensures the coherence of the paper.

B.2 WHY FOCUS ON THE RELATIONSHIP BETWEEN SEMANTIC SCALE AND ACCURACY?

It is important to note that we are concerned with the model bias introduced by unbalanced data, which causes models to perform poorly on some classes. In the past, researchers believed that models performed poorly on classes with fewer samples and therefore defined classes with fewer samples as tail classes and classes with more samples as head classes, proposing a long-tailed identification task. However, we observe that the model does not necessarily perform poorly on classes with fewer samples, which explains why some of the tail classes are "overbalanced" in many long-tailed identification methods. The higher similarity between our proposed semantic scale and model performance allows us to redefine the imbalance problem by replacing the number of samples with semantic scale. The superior performance achieved on the sample-balanced datasets shows that our proposed semantic scale imbalance is reliable. The semantic scale measure does not depend on the model and is calculated directly from the data. Even more surprising is that the semantic scale of the class can predict the performance of the class, which can lead to further understanding of what the model learns from the data. It can facilitate the development of data-driven artificial intelligence.

B.3 WHY SHOULD THE LOSS FUNCTION BE DYNAMICALLY WEIGHTED?

The working process of DSB loss: in each iteration, the semantic scale of each class is calculated in the feature space, and the loss function is re-weighted by the inverse of the semantic scale.

Why is the loss "dynamically" weighted? Because the semantic scales change continuously as the features change during training, and we need to update the features in each iteration and re-calculate the semantic scales to re-weight the loss function. The term "dynamic" refers to the dynamic update of the semantic scale in each iteration.

However, there is difficulty in implementing dynamic weighting, i.e., all features are needed to calculate the semantic scale, and we cannot extract the features of all samples in each iteration, which would be time consuming. Therefore, we analyze the "feature slow drift" phenomenon in Section 2 and propose to calculate the semantic scale by dynamically storing and updating the historical features (i.e., dynamic re-weighting training framework). Comparative experiments on the training speed and memory consumption of the above training framework are presented in Appendix F.2, and the results show that our approach is efficient.

C EXPERIMENTS ON STANFORD POINT CLOUD MANIFOLDS

Since $(Z - Z_{mean})(Z - Z_{mean})^T$ is a real symmetric matrix, it is semi-positive definite. Further, $I + \frac{p}{m}(Z - Z_{mean})(Z - Z_{mean})^T$ is a positive definite matrix and therefore $det(I + \frac{p}{m}(Z - Z_{mean})(Z - Z_{mean})^T) > 0$. The semantic scale measure is derived from the singular value decomposition of the data matrix, which is jointly determined by most of the samples. Therefore, our method is insensitive to noisy samples, i.e., the semantic scale measure is numerically stable. The semantic scales of multiple Stanford point cloud manifolds with different sizes are calculated and plotted in Figure 6. Let the center point of bunny be C_{bunny}. We increase the volume of bunny by performing $w * (bunny - C_{bunny})$, and the other point clouds are scaled up in this manner. As the object manifold is scaled up, the calculated volume then increases slowly and monotonically, indicating that our method can accurately measure the relative size of the manifold volume and is numerically stable, an advantage that will help mitigate the effects of noisy samples.

D EXPERIMENTAL DETAILS

D.1 MARGINAL EFFECT OF SEMANTIC SCALE

We use the following method to generate new training datasets for classification experiments: assume that the total number of classes in the original dataset is C, and m samples are randomly selected from each class to form a sub-dataset with a total number of samples of $C \times m$. The details of the sub-datasets generated based on CIFAR-10, CIFAR-100 and Mini-ImageNet are in Table 6.

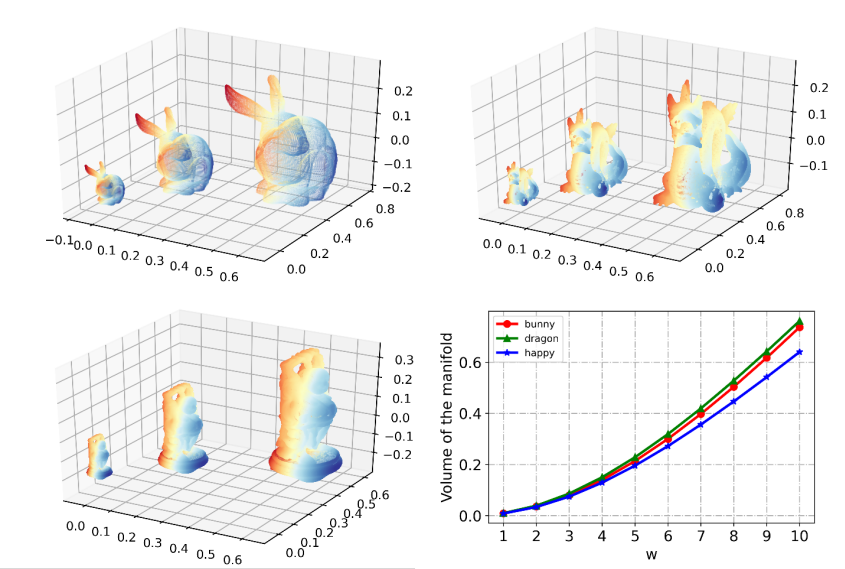


Figure 6: Increase three Stanford point cloud manifolds, and calculate their semantic scales.

Table 6: The sample-balanced sub-datasets with a total of 31. Among them, 13 sub-datasets are from CIFAR-10, 9 sub-datasets are from CIFAR-100, and the rest are from Mini-ImageNet. The test set remains the original test set. C denotes the total number of classes in the original dataset and m is the number of samples per class in the sub-dataset.

Dataset	C	Number	Sub-datasets												
CIFAR-10	10	m	50	100	200	300	400	500	600	800	1,000	2,000	3,000	4,000	5,000
		$C\times m$	500	1,000	2,000	3,000	4,000	5,000	6,000	8,000	10,000	20,000	30,000	40,000	50,000
CIFAR-100	100	m	10	20	30	50	100	200	300	400	500	-	-	-	-
		$C\times m$	1,000	2,000	3,000	5,000	10,000	20,000	30,000	40,000	50,000	-	-	-	-
Mini-ImageNet	100	m	10	20	30	50	100	200	300	400	500	-	-	-	-
		$C\times m$	1,000	2,000	3,000	5,000	10,000	20,000	30,000	40,000	50,000	-	-	-	-

D.2 Quantification of Semantic Scale Imbalance

We train ResNet-18 and ResNet-34 on CIFAR-10-LT and CIFAR-10 with an imbalance factor of 200, respectively, and the test set of CIFAR-10-LT is consistent with CIFAR-10. During training, the batch size is fixed to 64, and the optimizer adopts Adam. The learning rate is initially 0.01 and becomes $0.98\times$the previous learning rate after each epoch. We do not employ other additional tricks and data augmentation strategies.

D.3 Semantic Scale Imbalance on Long-Tailed Data

We artificially produce two long-tailed versions of the MNIST dataset, called MNIST-LT-1 and MNIST-LT-2. The number of samples per class is listed in Table 7.

Table 7: The two long-tailed MNIST datasets resampled from MNIST.

Dataset	MNIST-LT-1									
Class label	0	1	2	3	4	5	6	7	8	9
Number	5,923	3,590	2,940	2,518	2,256	1,972	1,700	1,300	1,100	900
Dataset	MNIST-LT-2									
Class label	0	1	2	3	4	5	6	7	8	9
Number	5,923	3,090	2,540	1,818	1,356	972	484	272	122	74

Figure 3 shows the class-wise accuracies of ResNet-18 and ResNet-34 trained on CIFAR-10-LT and CIFAR-100-LT with the same training settings as in Appendix D.2. Taking CIFAR-10 as an example, labels 1 to 10 correspond to: *airplane*, *automobile*, *bird*, *cat*, *deer*, *dog*, *frog*, *horse*, *ship*, *truck*. The prediction scores of classification experiments on CIFAR-10-LT and CIFAR-10 find that *cat* (label 4) is most easily confused with *dog* (label 6), as shown by their lowest accuracy on CIFAR-10 (Figure 4). However, the accuracy of *cat* and *dog* on CIFAR-10-LT is higher than that of *deer* (label 5), which is due to the dominant role of semantic scale S' in S for long-tailed data.

D.4 Semantic Scale Imbalance for More Datasets

We have demonstrated the semantic scale imbalance on MNIST, MNIST-LT, CIFAR-10, CIFAR-10-LT, CIFAR-100 and CIFAR-100-LT in Section 3.4. Figure 7 additionally shows the degree of semantic scale imbalance on CUB-2011, Cars196, and Mini-ImageNet, indicating that the semantic scale imbalance is indeed prevalent in all kinds of datasets.

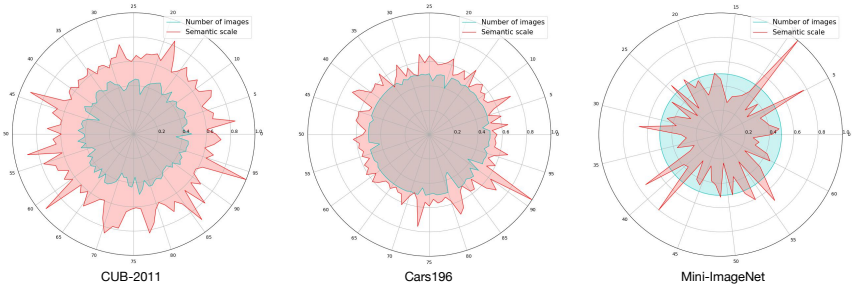


Figure 7: Semantic scale and number of samples per class. Different angles of the radar plot represent different classes, and the number of samples in the largest class is normalized to 0.5 for ease of observation.

D.5 Experimental Settings for Section 5.3 and More Experiments

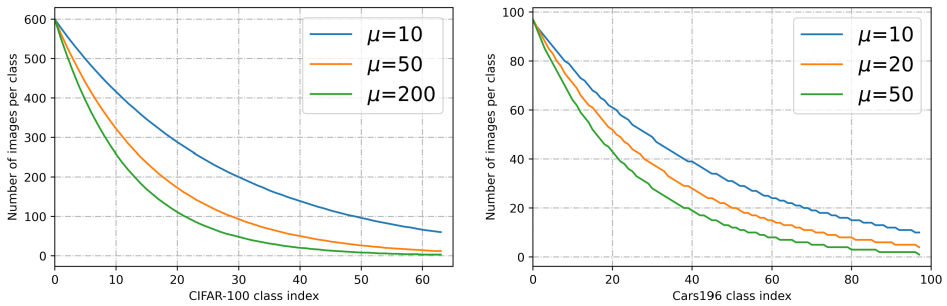
D.5.1 Experimental Settings for Section 5.3

Backbone Network and Experimental Parameters. Since we improve on the classical loss in the field of deep metric learning, we abide by the widely adopted backbone network, experimental parameters, and the division of datasets in this field. The BN-Inception (20; 47) pre-trained on ImageNet is adopted as the backbone network, and the training set is augmented by using random horizontal flipping and random cropping. All images are cropped to 224×224 as the input of the network. The output of the network after global average pooling is fed into a single fully connected layer to obtain 64- or 512-dimensional feature embeddings, and then all embeddings are clustered by K-means. The model is optimized by Adam with the batch size as 32 and the number of epochs as 50. We evaluate the performance of the learned embeddings with Recall@K and Normalized Mutual Information (NMI). The remaining experimental parameters used in the training are consistent with those reported in NormSoftmax and SoftTriple (37).

Dataset Introduction (CUB-2011, Cars196 and CIFAR-100-LT). The CUB-2011 dataset has 5,864 images in the first 100 classes for training and 5,924 images in the second 100 classes for testing. The Cars196 dataset consists of 196 classes totaling 16,185 images, with the first 98 classes for training and the remaining classes for testing. The CIFAR-100 has 100 classes, each containing 600 images. We create three long-tailed CIFAR-100 with the first 60 classes for training (See Figure 8a) and test on the remaining classes.

D.5.2 More Experiments

The long-tailed Cars196 is created using the first 98 classes for training (See Figure 8b) and the test set is the remaining classes. Both Mini-ImageNet and CIFAR-100 datasets contain 100 classes, each containing 600 samples. For the Mini-ImageNet dataset, the first 64 classes are the training set and the last 36 classes are the test set, and for the CIFAR-100 dataset, the first 60 classes are the



(a) (b)

Figure 8: Long-tailed CIFAR100 and Cars196 with different imbalance factors. We use the exponential function $n_i = N\mu^{i/(1-M)}$ to yield the number of training samples for each class, where i is the class index (0-indexed), N is the number of training samples in the largest class, μ is the imbalance factor, and M is the total number of classes.

training set and the remaining classes are the test set. Note that the experiments on CIFAR-100 in this section are different from the classification experiments on CIFAR-100 in Sec 5.3. The purpose of this experiments is to complement the effectiveness of our proposed method on both long-tailed and sample-balanced datasets for the field of deep metric learning.

Table 8: Comparison on long-tailed Cars196.

Dataset		Long-tailed Cars196								
Imbalance factor		10			20			50		
Metric		R@1	R@2	NMI	R@1	R@2	NMI	R@1	R@2	NMI
dim 64	NormSoftmax	66.4	76.9	58.9	63.1	74.2	54.9	59.5	71.1	52.9
	CB-NSM	68.9	78.0	60.1	64.9	75.2	56.1	61.7	72.5	53.3
	DSB-NSM	**69.5**	**78.6**	**60.7**	**65.4**	**75.7**	**56.6**	**62.3**	**73.0**	**54.1**
	SoftTriple	70.2	80.5	61.4	64.7	75.8	57.5	62.9	74.1	55.2
	CB-ST	71.9	81.3	62.9	66.5	76.9	58.6	64.8	75.4	56.0
	DSB-ST	**72.3**	**81.8**	**63.4**	**66.8**	**77.5**	**59.7**	**65.4**	**75.3**	**56.6**

Table 8 shows the performance comparison of DSB-NSM and DSB-ST on the long-tailed Car196. When the imbalance factor is 10, DSB-NSM outperforms NSM (NormSoftmax) by **3.1%** on R@1 and DSB-ST outperforms ST (SoftTriple) by **2.1%**. When the imbalance factor is 50, DSB-NSM and DSB-ST improve **2.8%** and **2.5%**, respectively, on R@1 compared to the original method. In addition, DSB loss performs better than CB loss on all metrics.

Table 9 shows that compared with the original losses, DSB-NSM and DSB-ST are able to consistently improve R@1 by **1.3%** on average and NMI by **1-2%** for both sample-balanced datasets, with all other metrics outperforming the original. The results in Tables 8 and 9 further confirm that our proposed dynamic semantic-scale-balanced learning is applicable to long-tailed and sample-balanced datasets in the field of deep metric learning, and has broad application prospects.

Table 9: Comparison on Mini-ImageNet and CIFAR-100.

Dataset	Mini-ImageNet			CIFAR-100		
Metric	R@1	R@2	NMI	R@1	R@2	NMI
NormSoftmax	85.7	91.2	74.1	60.1	71.5	49.4
DSB-NSM	**87.1**(+1.4)	**92.0**(+0.8)	**75.5**(+1.4)	**61.4**(+1.3)	**72.2**(+0.7)	**50.6**(+1.2)
SoftTriple	86.9	92.0	77.3	62.1	73.3	52.0
DSB-ST	**88.0**(+1.1)	**92.8**(+0.8)	**78.8**(+1.5)	**63.5**(+1.4)	**73.9**(+0.6)	**53.0**(+1.0).

E PSEUDO CODE FOR SAMPLE VOLUME

We provide the pseudo code for the calculation of the sample volume in Algorithm 1. In this work, we resize the image to (16, 16, 3) and then calculate the sample volume after flattening.

Algorithm 1 Calculation of Sample Volume

Input: Training set $D = \{(x_i, y_i)\}_{i=1}^{M}$ with the total number C of classes
Onput: Sample volumes for all classes
for $j = 1$ to C **do**
  Select the sample set $D_j = \{(x_i, y_i)\}_{i=1}^{m_j}$ for class j from D, m_j is the number of samples for class j
  Resize the image to $(imagesize, imagesize, 3)$
  Flatten the image into a vector of length $d = imagesize \times imagesize \times 3$ and store it in $Z_j = \left[z_1, z_2, \ldots, z_{m_j}\right] \in \mathbb{R}^{d \times m_j}$
  $Z_j = Z_j - NumPy.mean\left(Z_j, 1\right)$
  Calculate the covariance matrix $\Sigma_j = \frac{1}{m_j} Z_j Z_j^T$
  Calculate the sample volume $Vol\left(\Sigma_j\right) = \frac{1}{2}\log_2 \det\left(I + d\Sigma_j\right)$ for class j
end for

F DYNAMIC SEMANTIC-SCALE-BALANCED LEARNING

F.1 DSB-NSM, DSB-ST AND DSB-FOCAL LOSS

Given the embedding z of a sample and label y_i, the dynamic semantic-scale-balanced (DSB) loss can be expressed as:

$$DSB(z, y_i) = \frac{1}{S_i} L(z, y_i), i = 1, 2, \ldots, C, \tag{4}$$

where y_i is the label of the sample from class i. To show how to combine the general loss to generate the dynamic semantic-scale-balanced loss, we improve the NormSoftmax (NSM) cross-entropy loss and SoftTriple (ST) loss. NormSoftmax removes the bias term in the last linear layer and an L2 normalization module is added to the inputs and weights before the SoftMax loss. $[w_1, w_2, \cdots, w_C] \in \mathbb{R}^{d \times C}$ is the last fully connected layer, then the DSB-NSM with temperature σ generated by embedding z can be written as:

$$DSB-NSM(z, y_i) = -\frac{1}{S_i} \log\left(\frac{\exp(w_{y_i}^T z/\sigma)}{\sum_{j=1}^{C} \exp(w_j^T z/\sigma)}\right). \tag{5}$$

The SoftTriple loss combined with the semantic-scale-balanced term is expressed as

$$DSB-ST(z, y_i) = -\frac{1}{S_i} \log\left(\frac{\exp(\lambda(D'_{z,y_i} - \delta))}{\exp(\lambda(D'_{z,y_i} - \delta)) + \sum_{j \neq y} \exp(\lambda D'_{i,j})}\right), \tag{6}$$

where λ is a scaling factor and δ is a hyperparameter. The relaxed similarity between embedding z and class c is defined as $D'_{z,c} = \sum_k \frac{\exp(\frac{1}{\gamma} z^T w_c^k)}{\sum_k \exp(\frac{1}{\gamma} z^T w_c^k)} z^T w_c^k$, where k is the number of centers for each class.

The purpose of Focal loss is to apply small loss weights to samples with high classification confidence, thus increasing the proportion of loss of hard samples with low classification confidence to the overall loss. The α-balanced variant of Focal loss regulates the proportion of loss among samples while assigning different weights to each class, which is denoted as $FL\left(p_t\right) = -\alpha_t (1 - p_t)^{\gamma} \log\left(p_t\right)$, where p_t is the probability that the sample belongs to the true class. When $\alpha_t = \frac{1}{S_i}$, Focal loss is transformed into DSB-Focal loss.

F.2 DYNAMIC RE-WEIGHTING TRAINING FRAMEWORK

Given the training samples $X = [x_1, x_2, \ldots, x_N]$ containing C classes and corresponding labels $Y = [y_1, y_2, \ldots, y_N]$, the number of samples per class is $N_i\,(i = 1, 2 \ldots, C)$, and the total number of samples is N. The d-dimensional features extracted by the CNNs are denoted as

$Z = [z_1, z_2, \ldots, z_N] \in \mathbb{R}^{d \times N}$. In this work, we conduct experiments for two types of tasks, image classification and deep metric learning. In the field of deep metric learning, 64-dimensional features are generally adopted, while the features extracted by the network in image classification tasks tend to be of high dimensionality. For example, the feature dimension extracted by ResNet-50 is 2048, which will occupy more video memory. Therefore, when saving historical features in the classification task, one-dimensional average pooling is performed on all features to reduce the feature dimension to 64, which is consistent with the common feature dimension in the field of deep metric learning while preserving the geometry of the distribution (because the pooling operation is translation invariant, rotation invariant, and scale invariant). In the following, we describe the three-stage training framework in detail.

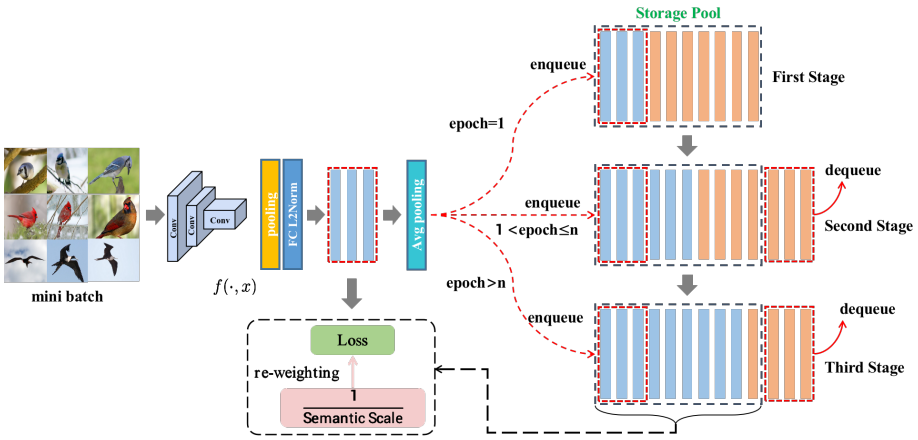


Figure 9: The three-stage training framework.

The three-stage training framework is shown in Figure 9:

(1) In the **first stage**, all the features and labels generated by the 1st epoch are stored in Q, which is denoted as:

$$Q = \begin{bmatrix} Z \\ Y \end{bmatrix} = \begin{bmatrix} Z_{11} & \cdots & Z_{1N} \\ \vdots & \ddots & \vdots \\ Z_{d1} & \cdots & Z_{dN} \\ y_1 & \cdots & y_N \end{bmatrix} \in \mathbb{R}^{(d+1) \times N}.$$

Q contains the features and labels of all samples, but in the early stage of training, the historical features have a large drift from the current features and cannot be used directly to calculate the semantic scale.

(2) The **second stage** corresponds to epoch 2 to epoch n. At each iteration, the oldest mini-batch features and labels in Q are removed and those generated by the current iteration are stored. The goal is to continuously update the features in Q until the feature drift is small enough. We set n to 5 in our experiments, and the original loss function is used in the first two stages. Figure 5 shows the effect of n on the model performance. A larger n does not hurt the model performance, but only takes a little more time. Experience suggests that setting n to 5 is sufficient.

(3) The **third stage** corresponds to epoch $> n$. At each iteration, the semantic scales are calculated using the features in Q after updating Q, and the original loss is re-weighted.

Algorithm 2 shows how to apply the dynamic re-weighting training framework by taking DSB-ST loss as an example.

The proposed three-stage training framework overcomes the difficulty of not being able to calculate class-wise semantic scales during training due to the limited number of samples per batch. In fact, there is a simple and brute-force method to achieve the goal of calculating class-wise semantic scales in real time, which is to extract features of all samples using the current model after each iteration. However, this would take a lot of time, for example, when training ImageNet with batch size of 512,

Algorithm 2 Dynamic Re-Weighting Training Framework

Input: Training set $D = \{(x_i, y_i)\}_{i=1}^{M}$, total training epochs N, defined encoder is model()
Initialize the queue Q
for $epoch = 1$ to N **do**
  for $iteration = 0$ to $\frac{M}{batchsize}$ **do**
    Sample a mini-batch $\{(x_i, y_i)\}_{i=1}^{batchsize}$ from D
    Calculate features $F = [f_1, \ldots, f_i, \ldots, f_{batchsize}]$, $f_i = \mathrm{model}\,(\mathrm{x_i})$, $\mathrm{i} = 1, \ldots,$ batchsize
    Store F and label y into Q: $enqueue\,(Q, [F, y])$
    if $epoch < n$ **then**
      if $epoch > 1$ **then**
        Dequeue the oldest mini-batch features from Q
      end if
      Calculate loss $L = SoftTripleloss\,(F, y)$
    else
      Dequeue the oldest mini-batch features from Q
      Calculate loss $L = DSB.SoftTripleloss\,(F, y)$
    end if
    Perform back propagation: L.backward()
    optimizer.step()
  end for
end for

one epoch contains about 2,500 iterations. This also means that all features need to be extracted 2,500 times, which is unacceptable.

Table 10: Comparison of DSB-ST and SoftTriple in terms of memory consumption and training speed. The speed is measured by the average number of iterations per second. The additional video memory consumption due to our method is almost negligible.

	GPU Memory		Training speed	
Dataset	SoftTriple	DSB-ST	SoftTriple	DSB-ST
ImageNet-LT	24.29 GB	24.75 GB	6.01 it/s	5.56 it/s
iNaturalist2018	45.13 GB	46.88 GB	3.32 it/s	3.03 it/s
Cars196	3491 MB	4097 MB	20.21 it/s	18.95 it/s
CUB-2011	3225 MB	3647 MB	21.95 it/s	18.58 it/s
Mini-ImageNet	3491 MB	3713 MB	23.34 it/s	20.36 it/s

Our method causes almost no reduction in training speed. In terms of video memory consumption, even the extracted features of a million-level dataset like ImageNet can all be placed on one graphics card (about 6,000 MB of video memory is needed). In this work, we use 4 NVIDIA 2080Ti GPUs to train all the models. A comparison of the video memory consumption and training speed for some experiments is shown in Table 10. It can be noticed that the consumption of our method is negligible for the video memory, and the training speed is about 90% of the original method.

G Volume Formula for the Low-Dimensional Parallel Hexahedron in the High-Dimensional Space

In Section 3.2 of the main content, we deduce from the singular value decomposition of the matrix $Z = [z_1, z_2, \ldots, z_m] \in \mathbb{R}^{d \times m}$ composed of features that the volume $Vol(Z)$ of the subspace spanned by z_i is proportional to $\sqrt{\det(\frac{1}{m} Z Z^T)}$. Here, we assume that in $\mathbb{R}^d$, given m d-dimensional vectors, these vectors will define a parallel hexahedron in $\mathbb{R}^n$. The problem is how to calculate the parallel

hexahedron. For example, consider two vectors

$$z_1 = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, z_2 = \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix}.$$

The parallel hexahedron defined by these two vectors is a parallelogram in $\mathbb{R}^3$. We want to find a formula to calculate the area of the parallelogram. (Note that the true three-dimensional volume of the planar parallelogram is 0, just as the length of the point is 0 and the area of the line is 0. Here, we are trying to measure the two-dimensional "volume" of the parallelogram.)

Next we will introduce two special cases of parallel hexahedral volume, for a single vector

$$z = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} \in \mathbb{R}^n,$$

whose parallel hexahedral is itself. Here "volume" means the length of the vector, and according to the Pythagorean theorem its volume is

$$\sqrt{a_1^2 + \cdots + a_n^2}. \tag{7}$$

Another case is to give n vectors in $\mathbb{R}^n$. Suppose that these n vectors are

$$z_1 = \begin{bmatrix} a_{11} \\ \vdots \\ a_{n1} \end{bmatrix}, \ldots, z_n = \begin{bmatrix} a_{1n} \\ \vdots \\ a_{nn} \end{bmatrix},$$

we can know that the volume of the resulting parallel hexahedron is

$$\left| \det \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \right|. \tag{8}$$

In the non-special case, the formula for the volume of a low-dimensional parallel hexahedron in a high-dimensional space will contain Results (7) and (8). Here, we first present the final formula and then discuss why it is reasonable. Write the k vectors $z_1, \ldots, z_k$ in $\mathbb{R}^n$ as column vectors. Let

$$Z = [z_1, \ldots, z_k] \in \mathbb{R}^{n \times k},$$

and the volume of the parallel hexahedron derived from the vectors $z_1, \ldots, z_k$ is

$$\sqrt{\det[Z^T Z]}.$$

We now discuss why $\sqrt{\det[Z^T Z]}$ must be the volume in the general case.

Lemma E.1. *For a matrix $Z = [z_1, \ldots, z_k]$, we can get*

$$Z^T Z = \begin{bmatrix} |z_1|^2 & z_1 \cdot z_2 & \cdots & z_1 \cdot z_k \\ \vdots & \vdots & \ddots & \vdots \\ z_k \cdot z_1 & z_k \cdot z_2 & \cdots & |z_k|^2 \end{bmatrix},$$

where $z_i \cdot z_j$ denotes the dot product of the vectors z_i and z_j and $|z_i| = \sqrt{z_i \cdot z_i}$ denotes the length of the vector.

The proof of Lemma 1 needs to focus only on

$$Z^T Z = \begin{bmatrix} z_1^T \\ \vdots \\ z_k^T \end{bmatrix} [z_1, \cdots, z_k].$$

If we apply any linear transformation that preserves angularity and length in $\mathbb{R}^n$ (in other words, if we perform a rotation operation on $\mathbb{R}^n$), the numbers $|z_i|$ and $z_i \cdot z_j$ do not change. The multiple sets of all linear transformations that preserve angle and length in $\mathbb{R}^n$ form a group, called the orthogonal group and denoted as $\mathrm{O}(n)$. This allows us to reduce the problem to that of finding the volume of a parallel hexahedron in $\mathbb{R}^k$.

Proof. It is known that

$$\sqrt{\det[Z^T Z]} = \sqrt{\det \begin{bmatrix} |z_1|^2 & z_1 \cdot z_2 & \cdots & z_1 \cdot z_k \\ \vdots & \vdots & \ddots & \vdots \\ z_k \cdot z_1 & z_k \cdot z_2 & \cdots & |z_k|^2 \end{bmatrix}}.$$

To prove that the above equation must be the formula for the volume, we first consider a set of standard basis of $\mathbb{R}^n$:

$$e_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, e_2 = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \ldots, e_n = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}.$$

According to Lemma 1, we are able to find a rotation of $\mathbb{R}^n$, which is able to maintain both length and angle, and also rotate our vectors $z_1, \ldots, z_k$ such that they can be fully represented linearly by the first k standard vectors $e_1, \ldots, e_k$ (which is geometrically reasonable). After the rotation, the latter $n-k$ dimensions of each vector zi are 0. Therefore we can think of our parallel hexahedron as consisting of k vectors in $\mathbb{R}^k$ and we already know how to calculate it, which is

$$\sqrt{\det \begin{bmatrix} |z_1|^2 & z_1 \cdot z_2 & \cdots & z_1 \cdot z_k \\ \vdots & \vdots & \ddots & \vdots \\ z_k \cdot z_1 & z_k \cdot z_2 & \cdots & |z_k|^2 \end{bmatrix}}.$$