Text-Supervised Local Feature Mixup Towards Long-Tailed Image Categorization

Richard Samuel Franklin

A thesis submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE AND SYSTEMS

University of Washington

2023

Reading Committee:

Juhua Hu, Chair

Qi Qian

Program Authorized to Offer Degree:
Computer Science and Systems

University of Washington

Abstract

Text-Supervised Local Feature Mixup Towards Long-Tailed Image Categorization

Richard Samuel Franklin

Chair of the Supervisory Committee:
Juhua Hu
School of Engineering and Technology

In many real-world applications, the frequency distribution of class labels for training deep visual models can exhibit a long-tailed distribution that challenges traditional approaches of training deep neural networks, which require heavy amounts of balanced data. Gathering and labeling data to balance out the class label distribution can be both costly and time-consuming. Many existing solutions that enable ensemble learning, re-balancing strategies, and fine-tuning applied to deep neural networks are limited by the inert problem of few class samples across a subset of classes. Recently, vision-language models like CLIP have been observed as effective solutions to zero-shot or few-shot learning by grasping a similarity between vision and language features for image and text pairs. Considering that large pre-trained vision-language models may contain valuable side textual information for minor classes, in this work, we propose to leverage text supervision to tackle the challenge of long-tailed learning for visual recognition. Furthermore, we propose a novel local feature mixup technique that takes advantage of the semantic relations between classes recognized by the pre-trained text encoder to further help alleviate the long-tailed problem. Our empirical study on several benchmark long-tailed tasks demonstrates the effectiveness of our proposal with a theoretical guarantee.

TABLE OF CONTENTS

Page

ACKNOWLEDGMENTS

This work would not be possible without the support of my advisor Dr. Juhua Hu and my parents for encouraging me to pursue my academic endeavors.

Chapter 1

INTRODUCTION

In recent years, deep learning has made state-of-the-art advancements in computer vision tasks such as image categorization, object detection, and semantic segmentation [47, 30]. Deep learning models are highly dependent on large-scale and balanced training data, but real-world data are typically class-imbalanced [31, 19, 3]. When training data is abundant for a subset of classes (i.e., head classes) but scarce for the other (i.e., tail classes), the distribution of the data is said to be long-tailed [53]. Taking image categorization as an example, deep neural networks (DNNs) aim to minimize the empirical risk on the training data by incrementally adjusting the learnable parameters. However, given a long-tailed training data, this happens more often on the head-class instances that appear more frequently, augmenting the model's performance bias towards head classes but reducing the model's generalization performance on tail classes [44, 3].

Long-tailed learning proves to be a significantly challenging task as addressed by many studies which have contributed to the field [44, 51, 55, 46, 34]. Intuitively, over-sampling the head and under-sampling the tail classes is a reasonable technique. Although class-level re-sampling or re-weighting can help to balance out the data distribution and mitigate the model's performance bias on head classes, these techniques can cause the model to overfit on the tail class set and reduce the performance on the head class set [38]. There is evidently more success in module improvement techniques [53, 20, 36], especially those that use ensemble learning [51, 44, 55]. There are a number of additional techniques [53] that aim to mitigate the long-tailed problem such as class-level re-margining [3], data augmentation [35, 6], and transfer learning [49]. However, these methods are still limited by the scarce information found among instances of the tail classes.

Recently, vision-language models such as CLIP [37] and ALIGN [21] have demonstrated good performance in zero-shot classification and few-shot learning [5]. These models are

trained on large-scale data containing image-text pairs that elicit the forming of connections between text and image embedding. By capturing the contrastive locality of image and text features, vision-language models can generalize to unseen categories well, which is a potential information source of tail classes in long-tailed learning. Therefore, in this work, we propose to leverage the pre-trained vision-language model that may contain additional information for tail classes in long-tailed learning.

Moreover, considering the observation that semantic relationships between class names (e.g., 'tiger' and 'cat') correlate with their localities of visual features in vision-language models, we can utilize semantically similar classes to assist the generalization among tail classes (e.g., the head class 'cat' can help assist the tail class of 'tiger' as shown in Fig. 3.2). However, the intra-class variance of the tail class can still be ignored. Therefore, we further propose a novel strategy, named local feature mixup (LFM), to shift the label towards tail classes, so as to alleviate the long-tailed problem. *This strategy distinguishes itself from pre-existing vision-language methods [42, 12, 22, 32] as LFM can not only be applied to vision-language architectures but also vision-only models due to its constraint on only fine-tuning the vision component.* Our extensive experiments on several benchmark long-tailed data demonstrate the effectiveness of our proposal. In this thesis, we aim to contribute to the field by proposing a novel method that takes advantage of vision-language model architectures.

Chapter 2

RELATED WORK

In long-tailed visual recognition, numerous experiments have been conducted to boost the performance of tail classes [53]. Module improvement methods including **ensembling** have shown recent success [51, 44, 55, 23]. In mixture of experts, TADE [51] and SHIKE [23] output an aggregation of multiple expert modules, where each expert in TADE strives to perform well in a different training distribution, and each expert in SHIKE focuses on modeling a different depth of image features. Although ensembling can boost performance, these methods are still limited by the scarce information found among images of the tail classes. **Data augmentation** techniques, which apply pre-defined image transformations, such as HOG [10] and SIFT [33], can be used to augment the tail class data. However, these methods lead to a limited performance boost, for they do not extract suitable high-level discriminative information from images [27].

One instance of success is found through **pre-training** vision transformers [13, 30] in an autoencoder setup [2, 25, 18, 39]. Once the encoder is sufficiently trained, it feeds into a classification layer that is trained using a balanced binary cross-entropy loss [3]. However, these methods still lack sufficient performance on the set of tail classes as it is an inert challenge to train deep neural networks with the limited information found in a few images.

Moreover, **class re-balancing** such as class-level re-sampling [9], re-weighting [8] (e.g. Balanced Cross Entropy [38]), and re-margining (e.g. LDAM [4]) can adjust the model's attention to classes with lower sample rates. However, class-balanced sampling or re-weighting can lead to overfitting of the tail classes and may under-represent the intra-class variance of the head classes [38, 53] thus decreasing the model's overall performance [41]. Alternatively, it can be effective to train a model with meta sampling [38] in which the optimal sample rate per class is estimated by applying a learnable parameter for each class label. Using this method can slightly avoid the overfitting of tail classes, but finding the optimal parameter

or trade-off between class labels for multi-class classification is difficult. Methods such as LDAM and meta-sampling solely rely on the class label's sample frequency in the training distribution to find the optimal tradeoff between classes. In this work, we discover a new trade-off by not only utilizing the class label frequency but also leveraging the semantic relationships between class labels via the text modality.

While the aforementioned techniques utilize the singular modality of images, we seek out an additional modality of text to extend the amount of information provided in the data. We leverage the text modality through the use of a **Contrastive Language-Vision Pre-training** (CLIP) [37] backbone. CLIP embodies multimodal learning through unsupervised training of image-caption pairs available on the wild web to capture the contrastive locality of image and text features. This makes them more adaptable to new and specific image classification tasks, so that they can be leveraged to make zero-shot predictions: generalize to unseen categories. A pre-trained multimodal model like CLIP or ALIGN [21] can be further fine-tuned on a downstream task to perform well in areas such as few-shot learning [1] and long-tailed recognition as in VL-LTR [42], LPT [12], VPT [22], and RAC [32].

The above works involve adapting both language and image modules of their respective networks to finetune their model on downstream tasks. Specifically, recent state-of-the-art VL-LTR requires manual retrieval of text descriptions of each class from the Internet to augment the text data in preparation for linguistic training, and this method currently outperforms all other vision-language methods within the domain. The following methods solely rely on the the internal training data. LPT and VPT tune their respective language modules through prompt tuning, and RAC transforms the text encoder output with a linear layer. We limit the scope by only finetuning the vision component and leveraging CLIP's text encoder, as is, to supervise the training of the image encoder making our method simple while also cutting on computational costs to train. Due to above reason, we do not include the mentioned language-trained methods in our evaluation for the sake of providing a fair comparison.

Chapter 3

THE PROPOSED METHOD

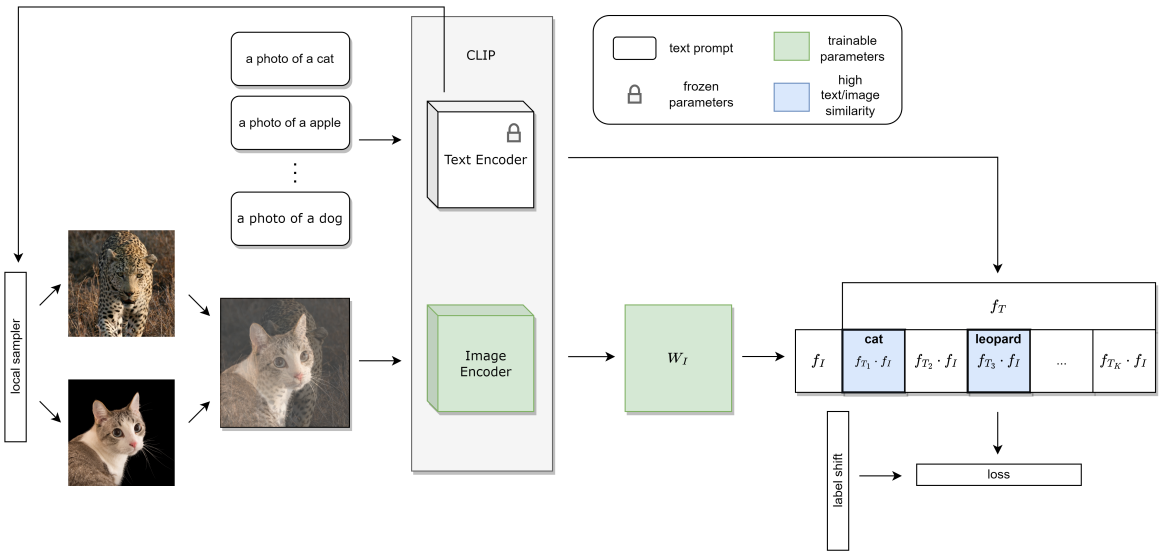


Figure 3.1: The proposed model architecture, in which the text encoder is fixed and LFM is applied using the pre-trained model by CLIP [37]. The image encoder will be fine-tuned according the downstream task and $W_I \in \mathbb{R}^{d \times d}$ is appended and learnable.

Given a long-tailed training data $D = \{(x_i, y_i)\}$, x_i is an image associated with its target class $y_i \in \{1, \ldots, C\}$. We construct a set of text snippets T, where each T_k describes a class label for $k \in \{1, \ldots, C\}$. For example, the text snippet describing class name "dog" is a tokenized sequence generated from the string, "a photo of a dog".

We feed image, x_i, and text snippets, T, to the image and text encoders, respectively, pre-trained by CLIP [37] as shown in Fig. 3.1, for which we denote as $\mathscr{F}_I$ and $\mathscr{F}_T$, respectively. Both of these encoders output feature vectors of size d. We denote the output from the text encoder as f_T and $f_T = \mathscr{F}_T(T)$ and allow f_{T_k} to denote the feature vector for class k, which

does not change during the long-tailed learning. To efficiently learn better presentations of images in a downstream image categorization task, we append a fully connected layer $W_I \in \mathbb{R}^{d \times d}$ that is learnable to $\mathscr{F}_I$. Thereafter, we can extract the feature vector for each image x_i as $f_I = W_I \mathscr{F}_I(x_i)$. Additionally, we normalize both f_{T_k} and f_I to be of a unit norm in the following.

After obtaining f_I and f_T, image classification is performed as shown in Fig. 3.1 by computing the cosine similarity between f_I and f_{T_k} for all k, and finally, the predicted class label, $\hat{y}$, for each image is computed as

$$\hat{y} = \underset{k \in \{1,\ldots,C\}}{\arg\max}\ f_I \cdot f_{T_k}\ .$$

This is distinguished from conventional classification predictions where a model is composed of a feature extractor and a classification layer, and a softmax of the logits are representative of a class prediction probability [17, 16, 30, 47, 13]. Moreover, by minimizing the empirical risk directly based on the training data with a long-trailed distribution, both $\mathscr{F}_I$ and W_I can be biased to the head classes. Therefore, to address the class imbalance problem, we propose a novel mixup technique that is supervised by the text encoder. Before introducing LFM, we provide the following overview of previous mixup techniques.

3.1 Preliminary Mixup Techniques

Previous mixup ideas exist as regularization and re-balancing techniques in training visual recognition models. Local feature mixup is similar and builds on top of these previous techniques.

3.1.1 Mixup

Mixup [50] was proposed as a data augmentation technique to improve the robustness of model predictions. Training samples $\tilde{x}$ and target $\tilde{y}$ are generated by the construct as

$$\tilde{x} = \lambda x_i + (1 - \lambda) x_j$$

$$\tilde{y} = \lambda y_i + (1 - \lambda) y_j$$

where (x_i, y_i) and (x_j, y_j) are randomly chosen from the training data, y_i, y_j are one-hot vectors, and $\lambda \in [0, 1]$ is randomly drawn from the beta distribution. At every iteration, the model is trained with $(\tilde{x}, \tilde{y})$ which is generated by a linear interpolation of samples from two different classes.

3.1.2 Remix

Rebalanced Mixup or Remix [6] was proposed as a re-balancing technique and can be directly used in long-tailed learning. Data point $(\tilde{x}, \tilde{y})$ is generated in a manner that favors model prediction of the tail classes. Similarly, (x_i, y_i) and (x_j, y_j) are randomly chosen from the training distribution where y_i, y_j are one-hot vectors, and we have:

$$\tilde{x}^{RM} = \lambda_x x_i + (1 - \lambda_x) x_j$$
$$\tilde{y}^{RM} = \lambda_y y_i + (1 - \lambda_y) y_j$$

where $\lambda_x \in [0, 1]$ is randomly chosen from the beta distribution and λ_y is defined as:

$$\lambda_y = \begin{cases} 0 & n_i / n_j \geq \kappa \text{ and } \lambda_x < \tau \\ 1 & n_i / n_j \leq 1/\kappa \text{ and } 1 - \lambda_x < \tau \\ \lambda_x & \text{otherwise} \end{cases}$$

where κ and τ are hyperparameters. Most importantly, Remix leads the model to predict in favor of classes with fewer sample rates. In other words, this technique pushes the decision boundary towards head classes and away from tail classes in anticipation of it fitting the balanced distribution more optimally. However, Remix is limited from utilizing multiple τ_{ij}, one for each class pair, because a desirable trade-off between classes is unknown. On the other hand, we have the opportunity to leverage CLIP's text encoder to retrieve a desireable trade-off: one that can be described as the semantic relationship between classes for each class pair. This motivates us to construct local feature mixup where the relationship between classes is automatically generated based on a provided text encoder's feature vector outputs.

3.2 Local Feature Mixup

A statistical measure of class imbalance in a dataset can be defined as the imbalance factor $\gamma = n_1/n_C$, where n_k is the number of examples in class k and $n_1 \geq n_2 \geq ... \geq n_C$, and typically, we have $n_1 \gg n_C$. Our main goal is to increase the few-shot accuracy (i.e., those with low n_k) while not overly attenuating the model's accuracy on many-shot classes (i.e., those with high n_k).

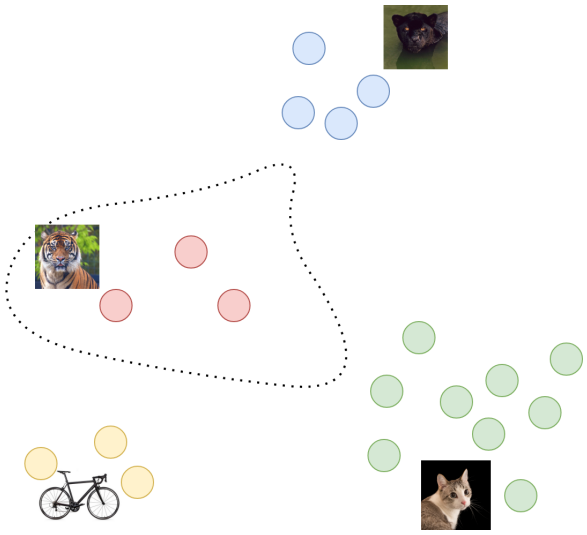


Figure 3.2: Local feature mixup allows semantically similar classes to be mixed more frequently (e.g., intra-class variance of 'tiger' as a tail class is stretched to the directions of 'leopard' and 'cat', where 'cat' as a head class can help).

We strive to boost the few-shot accuracy by making two assumptions about the data. First, we assume that classes with low n_k are underrepresented because a few examples may not fully express the complete diversity (or variance) of their associated class. For example, a cat can appear different from another cat in terms of their features such as their sizes, their eye colors, and the color and pattern of their furs. When limited to observing a few examples of cats, it is difficult for DNNs to grasp the full range of features that a cat can express, and thus, the model forms a limited concept of what cats look like. Therefore, we assume that every tail class has a larger intra-class variance than what can be learned from a long-tailed training data.

Secondly, because of the contrastive pre-training of CLIP [37], the text encoder already has an understanding of the local relationships between words. For example, words "frog" and "toad" are close in the language model feature space, since they have similar meanings. Part of our learning objective is to closely align the outputs of our image encoder to the outputs of the pre-trained language model. That is, if we feed an image of a frog and an image of a toad to our image encoder, their extracted feature vectors should be close in proximity as in the text feature space. Therefore, we also assume that if two classes have similar meanings (i.e., nearby in the text encoder's feature space), then these two classes also share a subset

of visual features and thus should also be nearby within the image encoder's feature space. To provide evidence towards this assumption, we make an observation (see Fig. 4.2) that zero-shot CLIP more frequently classifies images samples to be of nearby classes than that of distant classes.

In summary, we construct local feature mixup with *(1) the assumption that tail classes have larger intra-class variances than what can be learned from a long-tailed training dataset*, and *(2) the assumption that two classes that are nearby in the text encoder's feature space share a subset of visual features.* In the following construction of local feature mixup, we incorporate these two critical ideas separately, that is, local sampling and label shift.

3.2.1 Local Sampling

Existing mixup strategies often randomly sample y_i and y_j uniformly across the training data [50, 6, 43]. However, we aim to choose pairs that are related semantically that is supervised by the pre-trained text encoder. First, we sample an instance from class y_i uniformly across the training data as

$$p(y = y_i) = \frac{n_i}{\sum_k^C n_k} \tag{3.1}$$

Then, we sample another instance from class y_j with probability $p_{ls}(y = y_j|y_i)$ given by Eqn. 3.2. We emphasize that y_i and y_j are nearby in CLIP's text encoder feature space proportional to the cosine similarity of f_{T_i} and f_{T_j} as summarized in Alg. 1. By using this strategy, we hope to share the intra-class variance between nearby classes as our assumption is that they share a subset of visual features.

$$p_{ls}(y = y_j|y_i) = \begin{cases} \frac{exp(f_{T_i} \cdot f_{T_j}/\tau)}{\sum_{k=1}^C exp(f_{T_i} \cdot f_{T_k}/\tau)} & i \neq j \\ 0 & \text{o.w.} \end{cases} \tag{3.2}$$

where the hyperparameter $\tau > 0$ controls the temperature scaling on the softmax equation. A lower τ increases the likelihood that similar class pairs are chosen for mixup, but a too low temperature can lead to oversampling of nearby classes. We set $\tau = 0.05$ for most

experiments. Note that local sampling begets an auxiliary balancing effect on the training distribution as shown in Figs. 4.3-4.5.

Algorithm 1 LocalSample (τ, f_T, $D = \{(x, y)\}$)

$p_{y_i} \leftarrow [0, 1]^C$ vector representing the probability distribution from Eqn. 3.1
$p_{y_j|y_i} \leftarrow [0, 1]^{C\times C}$ matrix representing the probability distribution from Eqn. 3.2 with given τ and f_T
while model is not converged **do**
    $y_i \sim p_{y_i}$
    $y_j \sim p_{y_j|y_i}$
    $x_i \sim \{x \mid (x, y) \in D \text{ and } y = y_i\}$
    $x_j \sim \{x \mid (x, y) \in D \text{ and } y = y_j\}$
    yield (x_i, y_i), (x_j, y_j)
end while

3.2.2 Label Shift

Then, we perform mixup by mixing images x_i and x_j sampled through our above local sampling method. With mixing factors $\lambda_x, \lambda_y \in [0, 1]$, similar to the standard mixup [50] and Remix [6], we have

$$\tilde{x}^{LFM} = \lambda_x x_i + (1 - \lambda_x)x_j$$
$$\tilde{y}^{LFM} = \lambda_y y_i + (1 - \lambda_y)y_j$$

where y_i, y_j are one-hot vectors, and factor λ_x is also chosen randomly from the beta distribution. However, the difference lies in the way that we generate λ_y which is constructed by

$$\lambda_y = clamp(\lambda_x - \alpha\frac{n_i - n_j}{n_i + n_j}, 0, 1) \tag{3.3}$$

where the clamp function keeps λ_y between 0 and 1. In order to expand the margin for tail classes, we shift the decision boundary away from tail classes and towards head classes. The

(a) $\alpha = 0$ (b) $\alpha > 0$

Figure 3.3: An illustration of the theorized effect that label shift has on the model's decision boundary. Red circles indicate feature vectors of tail classes and green circles are that of nearby head classes. When $\alpha > 0$, the decision boundary shifts towards the head classes anticipating for higher intra-class variance for tail classes.

amount that is shifted depends on the difference of n_i and n_j, and this adjustment is scaled by hyperparameter $\alpha \geq 0$. For example, if $n_i > n_j$ (i.e., class y_i has more samples than class y_j), we shift the target to be more in favor of class y_j, thus increasing the model's margin on the class with fewer samples. It also follows that if $n_i < n_j$, we shift the target to be closer to class y_i. If $n_i = n_j$, $\lambda_y = \lambda_x$, which is equivalent to the original mixup technique. An illustration of our objective to shift the decision boundary away from tail classes can be observed in Fig. 3.3, which is also summarized in Alg. 2. The formal analysis can be shown as follows.

Theorem 1 *Letting* $p = n_i/(n_i + n_j)$, λ_y *can be obtained by balancing the distribution between* x_i *and* x_j

$$\lambda_y = \arg\min_{\lambda \in [0,1]} (\lambda - \lambda_x)^2/2 + \alpha R(\lambda)$$

where $R(\lambda) = (\lambda - 1/2)^2 - (\lambda - p)^2$.

Remark The former term constrains that the obtained weight for the label should be close to the weight for the example while the latter term is a balance regularization to incorporate the prior distribution p between two examples. By minimizing the regularization, it aims

to pushes λ from the imbalanced initial distribution to a balanced one. When $p = 1/2$, it degenerates to the standard weight for mixup.

Algorithm 2 Mix (α, (x_i, y_i), (x_j, y_j))

1: Convert y_i and y_j to one-hot vectors of size C
2: $\lambda_x \sim Beta(0.5, 0.5)$
3: $\lambda_y \leftarrow$ label shift assignment by Eqn. 3.3
4: $x^{LFM} \leftarrow \lambda_x x_i + (1 - \lambda_x) x_j$
5: $y^{LFM} \leftarrow \lambda_y y_i + (1 - \lambda_y) y_j$
6: **return** x^{LFM}, y^{LFM}

3.3 Training

We adopt a decoupled training approach as suggested by [34] in which training is performed in two phases, that is, phase 0 and phase 1. During phase 0, we continue to train the CLIP image encoder and update its weights, which we denote as Θ_I, and during phase 1, we freeze Θ_I and only update the fully connected layer W_I. In the beginning, W_I is initialized as the identity matrix, I_d, so that initially, $f_I = \mathscr{F}_I(x)$. During both phases of training, the feature vector outputs from the text and image encoders, f_T and f_I, are normalized to be unit vectors. We apply LFM in both phases to address the imbalance issue. Because we do not continue to train the text encoder, f_T is constant throughout the entire training procedure which limits the model's search space, reduces the computational cost when compared to other vision-language variants [42, 32, 12, 22], and makes our method more applicable to vision-only architectures.

Algorithm 3 Train ($\mathscr{F}_T$, $\mathscr{F}_I$, W_I, α, τ, D, T)

Initialize Θ_T and Θ_I (weights of $\mathscr{F}_T$ and $\mathscr{F}_I$, respectively) with pre-trained weights
Freeze Θ_T
$f_T \leftarrow \Pi_{||.||_2=1}\mathscr{F}_T(T)$
for epoch in $1, \ldots, N_0$ **do** ▷ Phase 0
    for $(x_i, y_i), (x_j, y_j)$ in LocalSample (τ, f_T, D) **do**
        $x^{LFM}, y^{LFM} \leftarrow Mix(\alpha, (x_i, y_i), (x_j, y_j))$
        $f_I \leftarrow \Pi_{||.||_2=1}\mathscr{F}_I(x^{LFM})$
        $\ell \leftarrow \mathcal{L}(f_T \cdot f_I, y^{LFM})$
        Update Θ_I
    end for
end for
Freeze Θ_I ▷ Phase 1
Initialize W_I as $d \times d$ identity matrix, I_d, where d is the feature dimension of $\mathscr{F}_I$
for epoch in $1, \ldots, N_1$ **do**
    for $(x_i, y_i), (x_j, y_j)$ in LocalSample (τ, f_T, D) **do**
        $x^{LFM}, y^{LFM} \leftarrow Mix(\alpha, (x_i, y_i), (x_j, y_j))$
        $f_I \leftarrow \Pi_{||.||_2=1}(W_I^T \mathscr{F}_I(x^{LFM}))$
        $\ell \leftarrow \mathcal{L}(f_T \cdot f_I, y^{LFM})$
        Update W_I
    end for
end for

Chapter 4

EXPERIMENTS

To demonstrate the proposed LFM method, we train the image encoder with LFM using imbalanced training data from the downstream tasks and evaluate performance on the corresponding balanced test. Following the common practice in long-tailed learning [53], we use publicly available long-tailed datasets, that is, CIFAR10-LT and CIFAR100-LT [26], ImageNet-LT [31], and Places-LT [54].

4.1 Experiment Setup

Table 4.1 details the hyperparameters and configuration properties used for each experiment. For CIFAR10/100-LT, we fine-tune CLIP with a single GPU, and for ImageNet-LT and Places-LT, we fine-tune CLIP with three GPUs. Each GPU is an Nvidia GeForce RTX 2080 Ti with 11GiB of memory. During training, each GPU receives a batch size of 32, so for ImageNet-LT and Places-LT the effective batch size is 96. Training is performed with a fixed seed to allow for reproduceability. Most experiments have the same setup, but some minor adjustments are made largely due to differences in class label distributions. Under the circumstances of heavy class imbalance, we can simply raise the values of α and τ, which we do for Places-LT [54]. Detailed information for each dataset is provided in Table 4.2. The original dataset imbalance is summarized by the imbalance factor γ, and γ' is the effective imbalance factor when we apply local sampling. A more detailed illustration of class label distributions for each dataset before and after applying our local sampling is provided in Figs. 4.3-4.5. Note that further hyperparameter tuning can help improve the model performance.

Table 4.1: Hyperparameters and configurations.

Dataset	CIFAR10-LT		CIFAR100-LT		ImageNet-LT		Places-LT	
Phase	0	1	0	1	0	1	0	1
Epochs	10	10	50	10	30	10	30	10
Learning Rate	1×10^{-9}	5×10^{-1}	1×10^{-6}	1×10^{-2}	5×10^{-6}	1×10^{-2}	1×10^{-7}	5×10^{-4}
LR Scheduler	Cosine Annealing		Cosine Annealing		Cosine Annealing		Cosine Annealing	
Min LR	1×10^{-12}	5×10^{-4}	1×10^{-9}	1×10^{-5}	5×10^{-9}	1×10^{-5}	1×10^{-10}	5×10^{-7}
Optimizer	Adam		Adam		Adam		Adam	
Batch Size	32		32		96		96	
α for LFM	1.00		1.00		1.00		1.25	1.50
τ for LFM	0.05		0.05		0.05		1.00	
Seed	0		0		0		0	

4.1.1 Text Prompting

Choosing the right text prompts, which we denote as T, can improve CLIP's classification accuracy. CLIP's default text prompt template is "a photo of a {CLASS}". For all experiments, we utilize the default text prompt template provided, considering the finding that a full sentence instead of a single word is generally more effective in describing the subject of the image [37].

4.1.2 Evaluation Metrics

A model's performance is not necessarily stable across all classes, each with different sample counts, so it is important that we quantify the performance of our model in subdivisions relative to every n_k. Across all datasets, we subdivide the resulting model's accuracy into four categories, namely many-shot, medium-shot, few-shot, and overall following [24]. Many-shot classes have $n_k > 100$, medium-shot classes have $20 \leq n_k \leq 100$, and few-shot classes have $n_k < 20$. For each performance category, we report the accuracy of our model against the balanced validation set for each subdivision of our chosen datasets. We additionally report the overall accuracy for all classes. This evaluation method of subdividing is frequently

Table 4.2: Detailed information of mentioned datasets

Dataset	CIFAR10-LT [26]			CIFAR100-LT [26]			ImageNet-LT [31]	Places-LT [54]
Number of classes	10			100			1000	365
Total Training Images	20,431	13,996	12,406	19,573	12,608	10,847	115,846	62,500
Max Images	5,000	5,000	5,000	500	500	500	1,280	4,980
Min Images	500	100	50	50	10	5	5	5
Original Imbalance Factor γ	10	50	100	10	50	100	256	996
Effective Imbalance Factor γ'	2.32	3.86	4.60	2.86	4.64	5.44	12.22	28.59

adopted by researchers working in the long-tailed learning for visual recognition [48].

In addition to reporting the accuracies of our model, we compare our proposed method with state-of-the-art baseline methods and strategies that perform well in tackling the long-tailed problem. We also fine-tune the competitive original CLIP's image encoder (i.e., ViT-B/32) with different existing losses as baselines, i.e., Cross Entropy (CE), Balanced Cross Entropy (BalCE) [38], Focal [28], Label Distribution Aware Margin (LDAM) [4], and Margin Metric Softmax (MMS) [40]. All losses except CE were proved to be helpful for the class imbalance problem. In summary, we compare with the following baselines based on the pre-trained CLIP [37].

- Zero-shot: The pre-trained image and text encoders by CLIP [37] are directly used to do prediction on the balanced test data, in which ViT-B/32 is adopted for the image encoder.

- CE: Fine-tuned ViT-B/32 from CLIP using the cross entropy loss.

- BalCE: Fine-tuned ViT-B/32 from CLIP using the balanced cross entropy loss [38].

- Focal: Fine-tuned ViT-B/32 from CLIP using the Focal loss [28].

- LDAM: Fine-tuned ViT-B/32 from CLIP using the loss in LDAM [4].

- MMS: Fine-tuned ViT-B/32 from CLIP with MMS [40].

Moreover, the performance reported in the original paper from existing state-of-the-art methods in long-tailed learning are also compared, e.g., BBN [55], LDAM [3], LiVT [46], RIDE [45], SHIKE [23], TADE [52], and GLMC [14].

4.2 CIFAR10/100-LT

As in the literature, we can create CIFAR10-LT and CIFAR100-LT by taking a subset of the original balanced CIFAR10 and CIFAR100 datasets [26], which can be executed by the imbalancer code [4] and the imbalance factor γ is variable. The long-tailed distribution of training data can be modeled by the exponential decay function. In other words, the new frequency per class, namely n_k is computed to be:

$$n_k = \bar{n} * \gamma^{-\frac{k}{C-1}}$$

where $\bar{n}$ is the balanced number of samples for every class in the original dataset. We experiment with multiple imbalance factors $\gamma = \{10, 50, 100\}$.

First, on CIFAR100 with the imbalance factor of 100, we compare all methods based on CLIP. For our proposal, we compare the performance of different backbones (i.e., ResNet50, ViT-B/32, and ViT-B/16) for the image encoder and apply LFM with two different losses, i.e., cross entropy and MMS [40] that is the best in the literature. Table 4.3 summarizes the results. Based on the zero-short performance, we can observe that the pre-trained CLIP can help balance the performance in different categories, which demonstrates the effectiveness of pre-trained vision-language model to alleviate the class imbalance issue. Then, by fine-tuning the pre-trained image encoder, the overall accuracy can be improved. However, due to the severe imbalance, the performance of the tail classes is still lacking even though balanced losses are utilized. Our proposal using ViT backbones can help improve the accuracy in all categories, where LFM combined with a loss well-suited for CLIP can further help improve the performance on the class imbalance problem.

Then, we compare the fine-tuned CLIP models (ViT-B/32 is adopted) including our proposal with multiple existing state-of-art long-tailed learning methods in Table 4.4 under different imbalance factors. We can observe that by fine-tuning the pre-trained CLIP image encoder, the performance can be significantly improved in all scenarios. Moreover, the

Table 4.3: CLIP-trained model accuracy on CIFAR100-LT with imbalance factor 100. The best is in bold and the 2nd best is underlined.

Methods	Backbone	Many	Med	Few	All
Zero-shot	ViT-B/32	63.5	60.8	61.4	62.0
CE	ViT-B/32	79.3	67.4	53.9	67.5
BalCE	ViT-B/32	74.6	69.8	57.4	67.6
Focal	ViT-B/32	80.2	65.0	54.0	66.9
LDAM	ViT-B/32	81.6	70.4	58.1	70.5
MMS	ViT-B/32	**90.3**	75.2	58.1	75.2
LFM + CE	ResNet50	55.5	59.4	48.4	54.6
LFM + CE	ViT-B/32	80.9	79.4	67.2	76.2
LFM + CE	ViT-B/16	83.4	<u>83.3</u>	72.8	<u>80.1</u>
LFM + MMS	ResNet50	51.4	53.6	43.6	49.7
LFM + MMS	ViT-B/32	79.5	81.1	**76.6**	79.2
LFM + MMS	ViT-B/16	<u>85.1</u>	**84.8**	<u>75.2</u>	**82.0**

state-of-the-art imbalance loss MMS [40] is very helpful, while our proposal can further significantly improve the performance in most cases. This further demonstrates the proposal of alleviating the class imbalance problem using pre-trained vision-language model and the effectiveness of LFM. It should be noted the backbone of ResNet50 is performing worse in general and is not adopted in the following experiments.

4.3 ImageNet-LT

We construct the long-tailed data distribution ImageNet-LT [31] by forming a subset of the ImageNet 2014 dataset [11]. This dataset contains over a million full resolution training images of 1000 common objects, where the objects of these images are nouns extracted from a subset of WordNet [15]. The resulting imbalance ratio of ImageNet-LT is 256. As shown in

Table 4.4: Overall accuracy on CIFAR10/100-LT with varying imbalance factors. The best is in bold and the 2nd best is underlined. '-' indicates that the accuracy is not available in the original paper.

Dataset	CIFAR10-LT			CIFAR100-LT		
Imbalance Factor	100	50	10	100	50	10
BBN [55]	79.8	82.2	88.3	42.6	47.0	59.1
LDAM [3]	77.0	-	88.2	42.0	-	58.7
LiVT [46]	86.3	-	91.3	58.2	-	69.2
RIDE [45]	-	-	-	48.0	51.7	61.8
SHIKE [23]	-	-	-	56.3	59.8	-
TADE [52]	-	-	-	49.8	53.9	63.6
GLMC [14]	87.75	90.22	94.04	57.11	62.32	72.33
CLIP Fine-tuning (ViT-B/32)						
CE	89.8	90.0	91.6	67.5	68.1	70.4
BalCE	91.3	91.6	92.4	67.6	68.8	70.8
Focal	89.8	90.0	91.6	66.9	68.6	70.4
LDAM	89.7	91.5	94.6	70.5	72.1	77.2
MMS	<u>93.3</u>	**94.5**	94.4	75.2	77.5	82.0
LFM + CE	**93.4**	<u>93.7</u>	**96.4**	<u>76.2</u>	<u>78.2</u>	<u>82.6</u>
LFM + MMS	89.1	91.0	<u>95.0</u>	**79.2**	**81.1**	**85.7**

Table 4.5, we can observe that compared to existing methods, CLIP methods beget better performance especially on few-shot accuracy (i.e., for tail classes), and our proposal can significantly improve the performance on all metrics.

Table 4.5: Performance comparison on ImageNet-LT. The best is in bold and the 2nd best is underlined. '-' indicates that the accuracy is not available in the original paper.

Methods	Many	Med	Few	All
CE [7]	64.0	33.8	5.8	41.6
LDAM [3]	60.4	46.9	30.7	49.8
LiVT [46]	<u>76.4</u>	59.7	42.7	63.8
RIDE [45]	68.3	53.5	35.9	56.8
SHIKE [23]	-	-	-	59.7
TADE [52]	66.5	57.0	43.5	58.8
GLMC [14]	70.1	55.9	45.5	57.21
CLIP Fine-tuning (ViT-B/16)				
Zero-shot	69.2	66.8	<u>65.8</u>	67.6
LFM + CE (ours)	69.8	**71.8**	**68.7**	<u>70.6</u>
LFM + MMS (ours)	**79.7**	<u>71.4</u>	51.3	**71.7**

4.4 Places-LT

In addition, we conduct experiments on Places-LT [31] using LFM with CE and MMS. Places-LT is a long-tailed subset of the original dataset Places2 [54]. It is a dataset for scene classification containing 365 classes, and it suffers from extreme imbalance ($\gamma = 996$). To account for its imbalance severity, we adjust local feature mixup hyperparameters to be highly in favor of the minority classes. We increase the value of τ, so that the probability distribution constructed by local sampling is more balanced. Additionally, we increase the value of α, so that the label is shifted to the tail classes, more heavily.

Table 4.6 summarizes the results. The benefit from the pre-trained model by CLIP can be observed from the zero-shot performance on tail classes, which further demonstrates the advantage of the text supervision from CLIP. However, fine-tuning using our proposal is necessary to improve the performance. It should also be noted that due to the severe

Table 4.6: Performance comparison on Places-LT. The best is in bold and the 2nd best is underlined. '-' indicates that the accuracy is not available in the original paper.

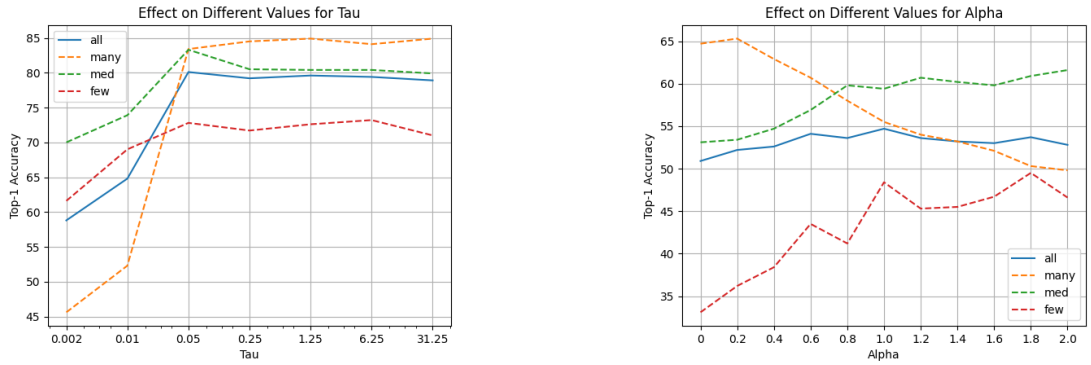
Methods	Many	Med	Few	All
CE [7]	<u>45.7</u>	27.3	8.2	30.2
Focal [29]	41.1	34.8	22.4	34.6
LiVT [46]	**50.7**	<u>42.4</u>	27.9	<u>42.6</u>
SHIKE [23]	43.6	39.2	44.8	41.9
TADE [52]	43.1	<u>42.4</u>	33.2	40.9
CLIP Fine-tuning (ViT-B/16)				
Zero-shot	36.8	35.8	45.1	38.1
LFM + CE (ours)	39.2	40.4	<u>46.4</u>	41.2
LFM + MMS (ours)	45.0	**48.1**	**46.7**	**46.7**

imbalance factor of this data, our proposal with CE is expected to be less effective compared to that with MMS [40]. LFM with MMS shows significantly better performance compared to state-of-the-arts, especially on medium-shot and few-shot classes, and demonstrates strong performance on many-shot classes as well. This further demonstrates the effectiveness of our proposed method on the long-tailed problem.

4.5 Ablation Studies

4.5.1 Analysis on Sample Probability

To study the effect of different temperature settings for p_{ls}, we run multiple experiments with $\tau = \{.002, .01, .05, .25, 1.25, 31.25\}$. At lower values, we increase the probability that nearby class samples (i, j) are paired together. At higher values, the probability of two nearby class samples becoming paired is mitigated, and the class sampling becomes more balanced. We run our experiments on CIFAR100-LT with an imbalanced factor of 100 using CLIP's ViT-B/16 backbone using the same configuration settings as observed in the supplementary.



(a) Effect on different τ

(b) Effect on different α

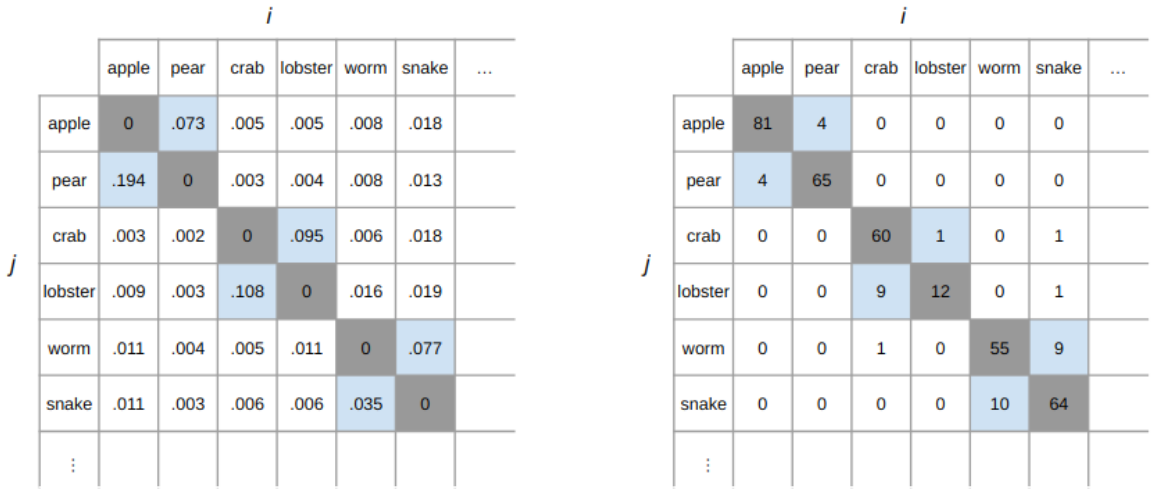
Figure 4.1: Ablation on CIFAR100-LT with imbalance factor 100.

Fig. 4.1a reveals that there is a sweet spot ($\tau = 0.05$) that works best for all accuracy, which is then fixed.

Additionally, Fig. 4.2 shows a comparison between the probability distribution for local feature mixup p_{ls} and a confusion matrix of CLIP's zero-shot classification performance using the CIFAR100's validation set. It can be observed that p_{ls} is correlated with the performance of zero-shot classification. By observing the blue cells in the confusion matrix, we can observe that the model more frequently struggles to find a decision boundary between related classes. With LFM, we expect that when we sample with p_{ls}, we share information with related (or "confused") classes more frequently and establish a decision boundary more optimally positioned for inference on the balanced validation data.

4.5.2 Analysis on Mixup Label Shift

We also study the effect of the intensity in which we shift the training label assigned to each mixup, for which we can control with α. The α value directly affects the positioning of the model's decision boundaries between class pairs, and we can expect lower values to extend the boundary of many-shot classes and higher values to extend the boundary of few-shot classes. In this study, we iterate α among the range $[0, 2]$ on CIFAR100-LT with



j \ i	apple	pear	crab	lobster	worm	snake	...
apple	0	.073	.005	.005	.008	.018	
pear	.194	0	.003	.004	.008	.013	
crab	.003	.002	0	.095	.006	.018	
lobster	.009	.003	.108	0	.016	.019	
worm	.011	.004	.005	.011	0	.077	
snake	.011	.003	.006	.006	.035	0	
⋮							

(a) $p_{ls}(y = y_j|y_i)$

j \ i	apple	pear	crab	lobster	worm	snake	...
apple	81	4	0	0	0	0	
pear	4	65	0	0	0	0	
crab	0	0	60	1	0	1	
lobster	0	0	9	12	0	1	
worm	0	0	1	0	55	9	
snake	0	0	0	0	10	64	
⋮							

(b) Confusion Matrix

Figure 4.2: The tables above demonstrate the correlation between the locality of text feature vectors and the model performance with zero-shot classification. The left matrix shows our constructed probability distribution p_{ls} for CIFAR100 and the right matrix is a confusion matrix of CLIP's performance on CIFAR100 without training. The columns represent class y_i, and the rows represent class y_j. For demonstration purposes, we choose three pairs of related classes to show: (apple, pear), (crab, lobster), and (snake, worm). Blue cells hold values for related class pairs while gray cells can be ignored since they hold the values for same class pairs. It can be observed that the blue cells hold values that are generally higher than any of the other white cells in their respective rows.

an imbalance factor of 100 using CLIP's ResNet50 backbone with the same configuration settings. From Fig. 4.1b, we can easily observe that an increasing of α can slowly degenerate the performance of many-shot classes while improve the performance of the other, especially that of the few-shot classes as expected. The result also reveals that setting α to 1 works best for all accuracies.

4.5.3 Analysis on Different Mixup Techniques

To demonstrate the proposed LFM, we also compare it with the standard Mixup [50] and Remix [6] on CIFAR100-LT. Specifically, we fine-tune CLIP with the same hyperparameters and decoupled stages, using different mixup techniques. Each model is trained using cross entropy loss with the ViT-B/16 backbone. Remix is a mixup method that addresses the class-imbalance issue, and it makes a trade-off between many-shot and few-shot performances. For example, compared to the standard mixup, Remix can help improve the performance on few-shot classes but sacrifice the performance on many-shot classes. However, Remix ignores the semantic relationship between each class pair that CLIP can be used for. Our proposal, which benefits from the semantic relationships between class pairs, shows significantly better performance in Table 4.7.

Table 4.7: Comparison of different mixup techniques on CIFAR100-LT with imbalance factor of 100

Methods	Many	Med	Few	All
Mixup [50]	80.4	71.5	55.1	69.5
Remix [6]	79.6	71.5	55.7	69.4
LFM (ours)	**83.4**	**83.3**	**72.8**	**80.1**

4.6 Visualized Effects of Our Method

4.6.1 Local Sampling Distribution

At each training step, local sampling feeds the model an image pair that holds semantically-related images, where the semantic relation is determined by the text encoder. In constructing the pair, the label of the first image is determined by uniformly sampling an image without replacement from the dataset, but the label of the second image is determined by Eqn. 3.2 that ignores the sample count for any class label. Due to the sampling method's negligence of the second label's sample count, the amount of times that the model sees minority classes

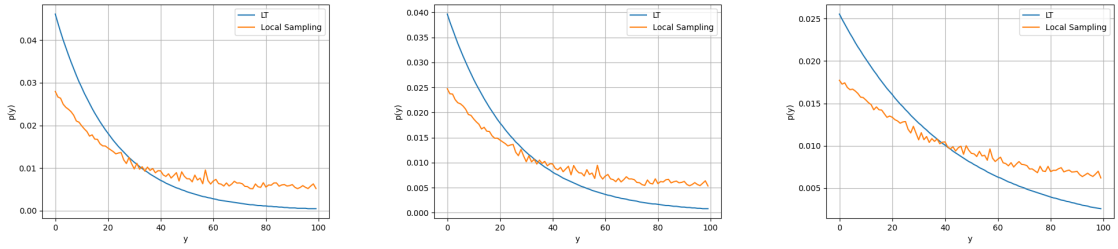
can be increased, effectively balancing the data distribution by resampling. To observe the amount of resampling that is done in local feature mixup, we show the sample count before and after local sampling as follows. Allow Y to be the random variable in the event that local sampling yields an instance of class $y \in \{y_i, y_j\}$, and allow y_i to be the event that $y_i = y$ and y_j to be the event that $y_j = y$. The probability that the model observes an image with class label y can be calculated as

$$\begin{aligned} p(Y = y) &= p(y_i) + (1 - p(y_i))p(y_j) \\ &= p(y_i) + (1 - p(y_i)) \sum_{k,k \neq i}^{C} p(y_j|y_k)p(y_k) \ . \end{aligned}$$

Using Eqns. 3.1 and 3.2, $p(Y)$ can be evaluated for all y, and we illustrate the resulting $p(y)$ for every dataset in Figs. 4.3-4.5. Additionally, we indicate the new imbalance factor as γ'. We can observe that our proposal can help effectively reduce the imbalance factor using our proposed local sampling method.

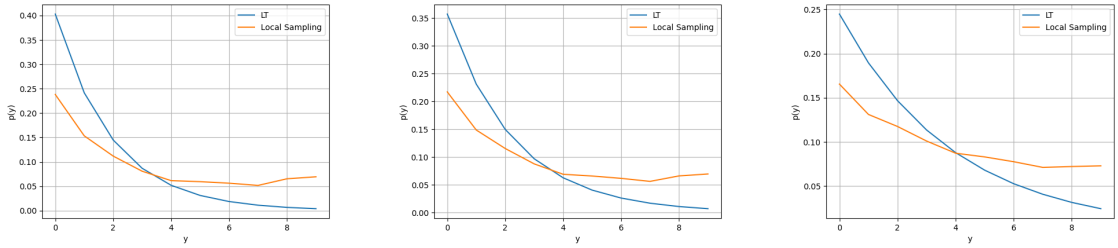
4.6.2 Image Feature Locality

To demonstrate our assumption on the local semantic relationship, we illustrate the geometric effect of fine-tuning CLIP with our proposed method in Figs. 4.6-4.7. We demonstrate the effect by revealing the contrastive locality of image feature vector outputs, where the input is comprised of a set of randomly sampled images from 10 chosen classes {apple, pear, ..., motorcycle} in CIFAR100. With the ViT-B/32 vision encoder and fully connected layer, we obtain a 512-dimensional feature vector for each image. To reduce the high-dimensional feature vectors to three dimensions (for human readability), we convert them using t-SNE trained for 1000 iterations and seed set to 1.



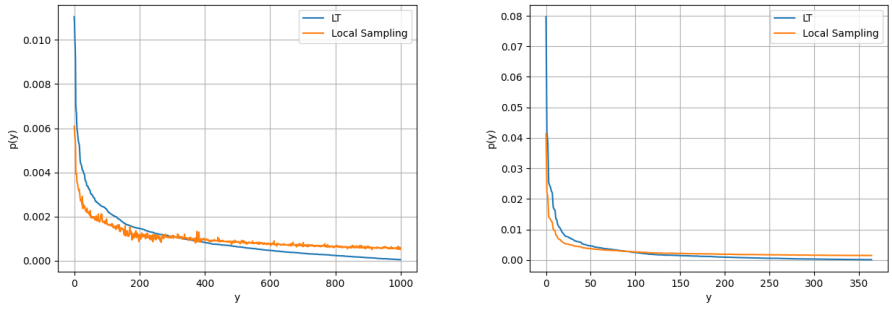
(a) $\gamma = 100$, $\gamma' = 5.44$ (b) $\gamma = 50$, $\gamma' = 4.64$ (c) $\gamma = 10$, $\gamma' = 2.86$

Figure 4.3: Local sampling effect on CIFAR100-LT $p(y)$ distribution



(a) $\gamma = 100$, $\gamma' = 4.60$ (b) $\gamma = 50$, $\gamma' = 3.86$ (c) $\gamma = 10$, $\gamma' = 2.32$

Figure 4.4: Local sampling effect on CIFAR10-LT $p(y)$ distribution



(a) ImageNet-LT ($\gamma = 500$, $\gamma' = 12.22$) (b) Places-LT ($\gamma = 996$, $\gamma' = 28.59$)

Figure 4.5: Local sampling effect on ImageNet-LT and Places-LT $p(y)$ distribution

(a) Zero Shot (b) CE (c) CE + LFM

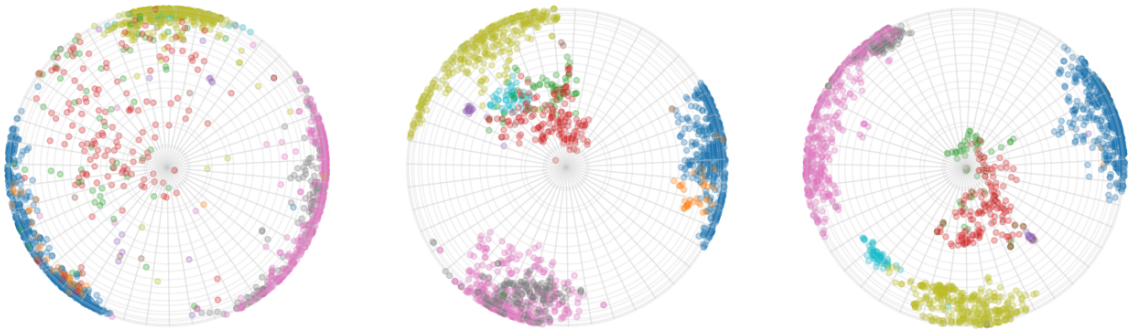


Figure 4.6: Top view

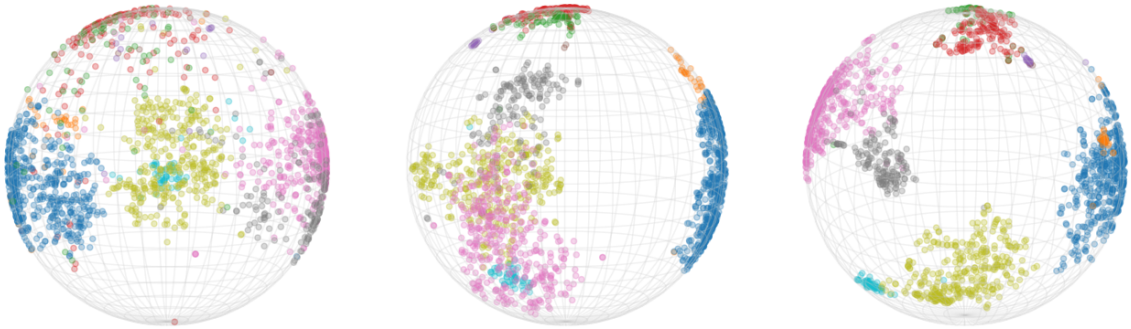


Figure 4.7: Front view

The legend contains the name of the class and the sample count in parenthesis. Our illustration contains the following 5 pairs of semantically related categories from CIFAR100: (apple, pair), (lobster, crab), (snake, worm), (bed, couch), and (bicycle, motorcycle). We choose these pairs to show that their semantic relations are aligned with their visual relations in terms of contrastive locality, as perceived by the image encoding layers. At zero shot, we can observe that semantically related classes are located nearby (e.g., 'apple' vs. 'pear' and 'bed' vs. 'couch'), although some are poorly clustered (e.g., 'lobster' vs. 'crab'). Fortunately, by incorporating our proposal of LFM, tail-class instances can be slightly separated from their semantically related head-class instances without sacrificing the clear boundaries between non-related

classes. For example, apple is similar to pear, so the pear cluster appears to be on the fringe of the apple cluster; (observable in Fig. 4.7); apple is not similar to bed and couch, so these clusters appear to be on opposite sides of the feature space. The class relationships become most apparent in plots generated by the CE + LFM model. For example, it is most apparent that LFM encourages the binding of class pairs: (bed, couch), (bicycle, motorcycle), and (apple, pear). This further demonstrates our proposal of leveraging a pre-trained vision-language model and LFM for long-tailed learning.

Chapter 5

CONCLUSION, LIMITATIONS, AND FUTURE WORK

Considering CLIP's ability to generalize to unseen categories, in this thesis, we propose to leverage a vision-language backbone to enhance the performance of image categorization over long-tailed training distributions. Furthermore, we propose a novel mixup technique that takes advantage of the semantic relationships between classes by probabilistic sampling classes based on their locality in the text encoder's feature space and slightly shifting the label towards tail classes. Our extensive experiments on several benchmark long-tailed training data demonstrate the effectiveness of our proposal in alleviating the class imbalance issue.

Although LFM is effective in multiple long-tailed evaluations, it has its own limitations. Firstly, it requires a language model pre-trained on a large language corpus to facilitate retrieval of class-wise semantic relationships, although it becomes very accessible recently. Secondly, LFM's performance degrades on datasets with a small amount of classes (see CIFAR10 results in Table 4.4) because LFM relies on sharing features between nearby classes via local sampling.

Numerous studies remain for future work. Firstly, we note that local feature mixup can not only be applied to vision-language architectures but also vision-only models due to our method's constraint on only finetuning the vision component. By applying LFM with other visual recognition tasks, we can assess the benefit of its application further. Secondly, there are other ways that we can leverage semantic relationships between the classes of our data. Our strategy may not only manifest as a mixup technique but also as a different type of data augmentation, logit adjustment, or consistency training technique. We may study these techniques in future experiments to further assess the benefits that language supervision has on the visual understanding in deep nueral networks. Lastly, both LFM and vision-language image classification are limited by the domain knowledge of the text encoder. Without further training, pretrained CLIP performs poorly on domain-specific tasks as suggested by

[42, 12, 22] due to its generic knowledge, e.g., for iNaturalist18 [19] that contains obscure class labels for species. Thus, in future work, we may study the effect of tuning the text encoder on the downstream task to enable the extraction of more informative features.

BIBLIOGRAPHY

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning, 2022.

[2] Dor Bank, Noam Koenigstein, and Raja Giryes. Autoencoders. *CoRR*, abs/2003.05991, 2020.

[3] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Aréchiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. *CoRR*, abs/1906.07413, 2019.

[4] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. In *Advances in Neural Information Processing Systems*, 2019.

[5] Jiaoyan Chen, Yuxia Geng, Zhuo Chen, Ian Horrocks, Jeff Z. Pan, and Huajun Chen. Knowledge-aware zero-shot learning: Survey and perspective. *CoRR*, abs/2103.00070, 2021.

[6] Hsin-Ping Chou, Shih-Chieh Chang, Jia-Yu Pan, Wei Wei, and Da-Cheng Juan. Remix: Rebalanced mixup. *CoRR*, abs/2007.03943, 2020.

[7] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples, 2019.

[8] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge J. Belongie. Class-balanced loss based on effective number of samples. *CoRR*, abs/1901.05555, 2019.

[9] Damien Dablain, Bartosz Krawczyk, and Nitesh V. Chawla. Deepsmote: Fusing deep learning and smote for imbalanced data. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–15, 2022.

[10] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, 2005.

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[12] Bowen Dong, Pan Zhou, Shuicheng Yan, and Wangmeng Zuo. Lpt: Long-tailed prompt tuning for image classification, 2023.

[13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020.

[14] Fei Du, Peng Yang, Qi Jia, Fengtao Nan, Xiaoting Chen, and Yun Yang. Global and local mixture consistency cumulative learning for long-tailed visual recognitions, 2023.

[15] Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. Bradford Books, 1998.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[17] Zhengyu He. Deep learning in image classification: A survey report. In *2020 2nd International Conference on Information Technology and Computer Application (ITCA)*, pages 174–177, 2020.

[18] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017.

[19] Grant Van Horn, Oisin Mac Aodha, Yang Song, Alexander Shepard, Hartwig Adam, Pietro Perona, and Serge J. Belongie. The inaturalist challenge 2017 dataset. *CoRR*, abs/1707.06642, 2017.

[20] Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. Learning deep representation for imbalanced classification. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5375–5384, 2016.

[21] Chao Jia and Yinfei Yang. Align: Scaling up visual and vision-language representation learning with noisy text supervision, May 2021.

[22] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning, 2022.

[23] Yan Jin, Mengke Li, Yang Lu, Yiu ming Cheung, and Hanzi Wang. Long-tailed visual recognition via self-heterogeneous integration with knowledge excavation, 2023.

[24] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. *CoRR*, abs/1910.09217, 2019.

[25] Diederik P. Kingma and Max Welling. An introduction to variational autoencoders. *CoRR*, abs/1906.02691, 2019.

[26] Alex Krizhevsky. Learning multiple layers of features from tiny images. pages 32–33, 2009.

[27] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[28] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection, 2018.

[29] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *CoRR*, abs/1708.02002, 2017.

[30] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *CoRR*, abs/2103.14030, 2021.

[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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[32] Alexander Long, Wei Yin, Thalaiyasingam Ajanthan, Vu Nguyen, Pulak Purkait, Ravi Garg, Alan Blair, Chunhua Shen, and Anton van den Hengel. Retrieval augmented classification for long-tail visual recognition, 2022.

[33] David Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–, 11 2004.

[34] Teli Ma, Shijie Geng, Mengmeng Wang, Jing Shao, Jiasen Lu, Hongsheng Li, Peng Gao, and Yu Qiao. A simple long-tailed recognition baseline via vision-language model. *CoRR*, abs/2111.14745, 2021.

[35] Agnieszka Mikołajczyk and Michał Grochowski. Data augmentation for improving deep learning in image classification problem. In *2018 International Interdisciplinary PhD Workshop (IIPhDW)*, pages 117–122, 2018.

[36] Wanli Ouyang, Xiaogang Wang, Cong Zhang, and Xiaokang Yang. Factors in finetuning deep model for object detection with long-tail distribution. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 864–873, 2016.

[37] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021.

[38] Jiawei Ren, Cunjun Yu, Shunan Sheng, Xiao Ma, Haiyu Zhao, Shuai Yi, and Hongsheng Li. Balanced meta-softmax for long-tailed visual recognition. *CoRR*, abs/2007.10740, 2020.

[39] Dvir Samuel, Yuval Atzmon, and Gal Chechik. Long-tail learning with attributes. *CoRR*, abs/2004.02235, 2020.

[40] Yang Shu, Xingzhuo Guo, Jialong Wu, Ximei Wang, Jianmin Wang, and Mingsheng Long. Clipood: Generalizing clip to out-of-distributions, 2023.

[41] Saptarshi Sinha, Hiroki Ohashi, and Katsuyuki Nakamura. Class-wise difficulty-balanced loss for solving class-imbalance. *CoRR*, abs/2010.01824, 2020.

[42] Changyao Tian, Wenhai Wang, Xizhou Zhu, Jifeng Dai, and Yu Qiao. Vl-ltr: Learning class-wise visual-linguistic representation for long-tailed visual recognition, 2022.

[43] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, Aaron Courville, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states, 2018.

[44] Xudong Wang, Long Lian, Zhongqi Miao, Ziwei Liu, and Stella X. Yu. Long-tailed recognition by routing diverse distribution-aware experts. *CoRR*, abs/2010.01809, 2020.

[45] Xudong Wang, Long Lian, Zhongqi Miao, Ziwei Liu, and Stella X. Yu. Long-tailed recognition by routing diverse distribution-aware experts, 2022.

[46] Zhengzhuo Xu, Ruikang Liu, Shuo Yang, Zenghao Chai, and Chun Yuan. Learning imbalanced data with vision transformers, 2022.

[47] Jianwei Yang, Chunyuan Li, Xiyang Dai, Lu Yuan, and Jianfeng Gao. Focal modulation networks, 2022.

[48] Lu Yang, He Jiang, Qing Song, and Jun Guo. A survey on long-tailed visual recognition. *International Journal of Computer Vision*, 130(7):1837–1872, may 2022.

[49] Xi Yin, Xiang Yu, Kihyuk Sohn, Xiaoming Liu, and Manmohan Chandraker. Feature transfer learning for deep face recognition with long-tail data. *CoRR*, abs/1803.09014, 2018.

[50] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *CoRR*, abs/1710.09412, 2017.

[51] Yifan Zhang, Bryan Hooi, Lanqing Hong, and Jiashi Feng. Test-agnostic long-tailed recognition by test-time aggregating diverse experts with self-supervision. *CoRR*, abs/2107.09249, 2021.

[52] Yifan Zhang, Bryan Hooi, Lanqing Hong, and Jiashi Feng. Self-supervised aggregation of diverse experts for test-agnostic long-tailed recognition, 2022.

[53] Yifan Zhang, Bingyi Kang, Bryan Hooi, Shuicheng Yan, and Jiashi Feng. Deep long-tailed learning: A survey. *CoRR*, abs/2110.04596, 2021.

[54] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[55] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. BBN: bilateral-branch network with cumulative learning for long-tailed visual recognition. *CoRR*, abs/1912.02413, 2019.