From Expansion to Retraction: Long-tailed Machine Unlearning via Boundary Manipulation

Min Chen
Hubei Key Laboratory of Internet of Intelligence, School of EIC, Huazhong University of Science and Technology
Wuhan, China
chenmin7@hust.edu.cn

Weizhuo Gao
Hubei Key Laboratory of Internet of Intelligence, School of EIC, Huazhong University of Science and Technology
Wuhan, China
wzgao@hust.edu.cn

Chen Wang
Hubei Key Laboratory of Internet of Intelligence, School of EIC, Huazhong University of Science and Technology
Wuhan, China
chenwang@hust.edu.cn

Gaoyang Liu*
Hubei Key Laboratory of Internet of Intelligence, School of EIC, Huazhong University of Science and Technology
Wuhan, China
liugaoyang@hust.edu.cn

Ahmed M. Abdelmoniem
School of Electronic Engineering and Computer Science, Queen Mary University of London
London, UK
ahmed.sayed@qmul.ac.uk

Kai Peng
Hubei Key Laboratory of Internet of Intelligence, School of EIC, Huazhong University of Science and Technology
Wuhan, China
pkhust@hust.edu.cn

Abstract

Machine unlearning aims to remove the information of specific data from a trained machine learning model while retaining its utility for the remaining data, so as to meet the requirements of privacy regulations. Existing unlearning methods often assume a balanced data distribution, but neglect the real-world, long-tailed scenarios, where the decision boundaries of tail classes are frequently distorted due to insufficient sample representation, thereby reducing the unlearning efficacy. In this paper, we propose the first Long-Tailed Machine Unlearning (LTMU) framework from a unified decision-boundary perspective. Our framework begins with a directional boundary repair scheme designed to enrich the distorted decision boundary of the tail class, and then develop a novel boundary retraction approach tailored for long-tailed unlearning, dispersing both the augmented and original features throughout the feature space. This bidirectional manipulation not only offers a unified interpretation of the relationship between long-tailed learning and unlearning, but also enables flexible control over both repair and unlearning processes through the generation of augmented features, thereby effectively accomplishing the long-tailed unlearning task. Extensive experiments across multiple datasets and neural network architectures demonstrate the effectiveness of our framework in achieving complete unlearning of tail classes in long-tailed distributions.

CCS Concepts

• **Computing methodologies** → **Machine learning**; • **Security and privacy** → **Information accountability and usage control**; **Usability in security and privacy**.

Keywords

Machine unlearning, long-tailed learning, directional boundary repair, boundary retraction.

ACM Reference Format:
Min Chen, Weizhuo Gao, Chen Wang, Gaoyang Liu, Ahmed M. Abdelmoniem, and Kai Peng. 2025. From Expansion to Retraction: Long-tailed Machine Unlearning via Boundary Manipulation. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (KDD '25), August 3–7, 2025, Toronto, ON, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3711896.3736970

1 Introduction

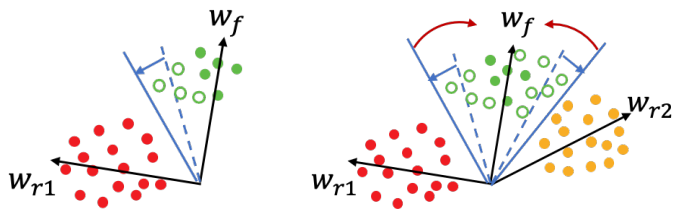
Machine unlearning focuses on the removal of specific information from a well-trained machine learning model while preserving its utility on the remaining data [38]. The need for unlearning arises from various concerns, including privacy regulations such as GDPR [34] and CCPA [10], security vulnerabilities like model poisoning attacks [25], and the requirement to eliminate outdated or erroneous data from models to maintain their relevance and accuracy [13]. As a result, machine unlearning has garnered great research interests in recent years.

However, existing machine unlearning methods often rely on the assumption that the training data follows a balanced distribution. This assumption, however, may not hold in some practical scenarios like face attribute prediction and person re-identification [19], where imbalanced data distributions are common. Such imbalance weakens the generalization capabilities of classes represented by

*Corresponding author. This work was supported in part by the National Natural Science Foundation of China under Grants 62272183, 62171189 and 62071192; by the National Key R&D Program of China under Grant 2024YFE0103800; by the Major Science and Technology Project of Hubei Province under Grants 2024BAA008 and 2024BAA011; by the Key R&D Program of Hubei Province under Grants 2024BAB016, 2024BAB031 and 2023BAB074; and by the UKRI EPSRC Grant EP/X035085/1.

KDD '25, Toronto, ON, Canada

ACM ISBN 979-8-4007-1454-2/2025/08
https://doi.org/10.1145/3711896.3736970



(a) Repairing decision boundary **(b) Boundary manipulation**

Figure 1: Illustration of decision boundary manipulation in long-tailed learning and unlearning. (a) Long-tailed learning techniques aim to repair (blue arrows) the distorted boundary (blue dashed lines) of tail classes (green dots) by generating augmented features (green circles). (b) Unlearning is achieved by first expanding the boundary (blue solid lines) and then dispersing (red arrows) the feature space of the tail class. Best viewed in color.

fewer samples, thereby limiting the efficacy of current unlearning methods in removing data associated with these tail classes[1].

To address the issue of imbalanced training data distribution, information augmentation has been widely adopted in long-tailed learning, which generates additional augmented features by fitting the statistical distribution of the tail classes [41]. Since insufficient features may fail to adequately represent the full feature space of the tail classes, these augmented features are then used to repair the distorted decision boundaries of the tail classes during the original training phase (c.f. Figure 1a).

However, current augmentation approaches [28, 41] often overlook the intrinsic similarities between tail classes and their neighboring classes [6], and generate more augmented samples for tail classes. This inconsiderate augmentation process risks blurring the decision boundaries of the remaining classes, which would potentially compromise their remaining data's utility in the unlearned model. For instance, in Figure 1b, class f is more semantically similar to class $r2$ than to class $r1$, suggesting that the augmented features of f near $r2$ should be more compact than on other sides. Ignoring this inter-class similarity may cause these augmentations for the tail class (green circles) to encroach upon the feature space of neighboring classes (yellow dots). Consequently, current augmentation methods may hinder the preservation of remaining data utility, which is a key objective in unlearning task.

In this paper, we propose the Long-Tailed Machine Unlearning (LTMU) framework from a unified decision-boundary perspective, by first expanding the decision boundary outward in the repair phase, and then contracting it inward during the unlearning phase (c.f. Figure 1b). Specifically, our framework begins with a directional boundary repair scheme designed to enrich the distorted decision boundary of the tail class, guided by intrinsic inter-class similarities. This scheme carefully preserves similarity relationships among neighboring classes by identifying the top k most similar classes and generating augmented features directed toward them. These tailored augmentation features can thus efficiently and accurately enhance the representational capacity of the tail classes.

On this basis, we further propose a novel boundary-retraction approach tailored for long-tailed unlearning, by retracting the decision boundary of the tail class away from the nearest neighboring classes. In contrast to prior unlearning method [6], which use adversarial labels to guide boundary retraction, our approach leverages the inter-class similarity calculated during the repair phase to disperse both the augmented and original features throughout the feature space, offering greater time efficiency. By doing so, our framework can unify the repair and unlearning processes into a bidirectional outward-inward manipulation of the decision boundary throughout the feature space. This bidirectional manipulation not only offers a unified interpretation of the relationship between long-tailed learning and unlearning, but also enables flexible control over both processes through the generation of augmented features, thereby effectively accomplishing the long-tailed unlearning task.

Our major contributions can be summarized as follows:

- We introduce LTMU, the first unlearning framework for removing a tail class from a deep model trained on imbalanced data. Through a tailored bidirectional boundary manipulation, our approach can fully erase the tail class while maintaining the utility of the remaining classes.
- We design a directional boundary repair scheme by generating augmented features of the tail class that can preserve the intrinsic inter-class similarity. The generated features can be then used to expand outward the decision boundary of the tail class, yielding greater representation diversity of the tail class.
- We develop a boundary retraction approach, by retracting the decision boundary of the tail class inward the nearest classes, to efficiently and effectively disperse the tail class features.
- Extensive experiments on four datasets across various neural network architectures validate the proposed LTMU framework, demonstrating its effectiveness in unlearning tail classes under long-tailed distribution settings. Our code is available at https://github.com/SPHelixLab/LTMU

2 Related Work

2.1 Machine Unlearning

Machine unlearning techniques are broadly divided into certified and approximate approaches [38]. Certified unlearning ensures that the parameter distribution in the unlearned model is indistinguishable from that of the retrained model, thus guaranteeing strong theoretical correctness. Early works focus on convex models [2, 4, 14, 17], but struggle to scale to deep neural networks. Recent efforts like SISA [1] and Amnesiac Unlearning [16] optimize exact unlearning for deep models, using partitioned sub-models or parameter tracking to remove specific training information. However, these approaches disrupt the original training pipeline, require extensive memory, and may compromise model utility.

Approximate unlearning methods aim for inference behaviors in the unlearned model to approximate those in a retrained model. Recent approaches leverage neural network characteristics to achieve efficient unlearning. For example, Fisher forgetting [15] identifies

[1]Categories with sufficient samples are referred to as head classes, while those with few samples are known as tail classes.

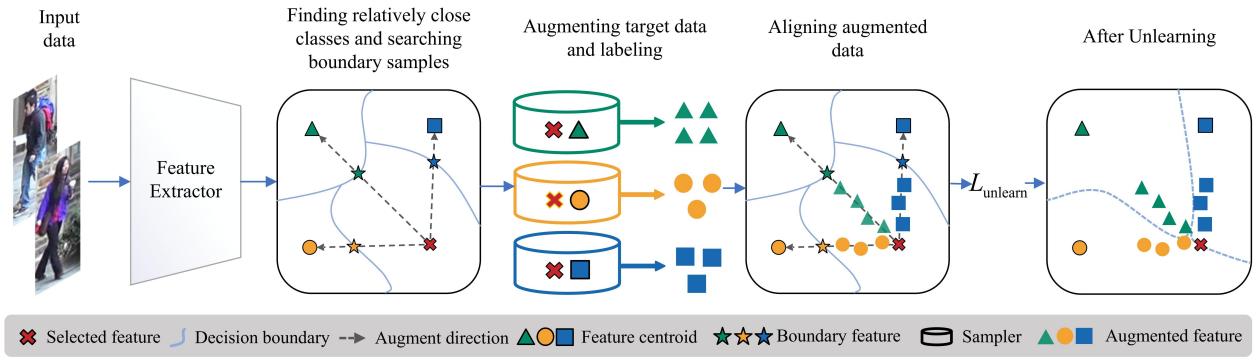


Figure 2: Schematic overview of the proposed LTMU framework, illustrating the key stages: identifying boundary samples, generating directional augmented features, and performing boundary repair and retraction. The augmented features are used to repair the distorted decision boundary of the tail class, while the combination of augmented and original features enables boundary retraction, ensuring comprehensive unlearning of the tail class.

influential parameters for forgetting data and alters them with noise. SCRUB [22] enforces a constraint on the KL-divergence between the output distributions of a student model and a teacher model for both the remaining and forgetting data, updating the parameters accordingly to achieve unlearning. UNSC [5] employs null-space projection to isolate forgetting information, while Boundary Unlearning [6] compresses the decision space of the forgetting class. Similarly, L1-sparse unlearning [24] uses model sparsity to simplify forgetting by fine-tuning with l_1 regularization, and SalUn [12] applies hard thresholding to erase sensitive information.

Nevertheless, nearly all existing unlearning methods are built on the assumption of balanced datasets, making them challenging to apply in long-tailed settings.

2.2 Boundary Repairing Methods

Boundary repair techniques for long-tailed learning share a similar augmentation concept of generalization ability of tail class with our long-tailed unlearning task. For instance, Yin *et al.* [39] generate additional features for tail classes by leveraging variation information from head classes, and then optimize the model with these additional tail features to reduce the decision boundary bias. Similarly, RSG [35] proposes using a fully parameterized layer to capture variation information, enabling the generation of more stable and representative augmented samples for tail classes. Ma *et al.* [28] focus on transferring the variance from head classes, estimating multiple sub-distributions centered on each tail class feature to more accurately represent the tail class. Furthermore, FASA [41] generates tail features based on a distributional prior, with statistics calculated from previously observed tail samples. Recently, Ma *et al.* [27] investigate the impact of the curvature of perceptual manifolds, which represent second-order properties of the decision boundary [29], to analyze model bias in long-tailed settings.

While these methods generate augmented tail features to repair the distorted decision boundaries of tail classes, they lack the ability to capture inter-class similarity, which is crucial for preserving the utility of the remaining data in the unlearning context. Moreover, integrating existing boundary repair methods with most unlearning approaches is challenging, as these methods either do not utilize the forgetting data or do not require the computation of the feature space. This highlights the need for our unified boundary-based framework. Our LTMU integrates repair and unlearning as manipulations of the decision boundary across the feature space, which can achieve more efficient long-tailed machine unlearning performance.

3 Method

We consider long-tailed machine unlearning within the context of supervised classification using deep networks. Let $\mathcal{D} = (\boldsymbol{x}_i, \boldsymbol{y}_i)_{i=1}^N \subseteq \mathcal{X} \times \mathcal{Y}$ represent the training dataset used to train the original model $\mathcal{M}_o$. The label space $\mathcal{Y} = \{1, ..., C\}$ consists of C categories. Our aim is to remove the information of forgetting data $\mathcal{D}_f \subset \mathcal{D}$ from the original model $\mathcal{M}_o$, while retaining the knowledge of remaining data $\mathcal{D}_r = \mathcal{D} \setminus \mathcal{D}_f$. The unlearned model $\mathcal{M}_u$ is expected to closely resemble the retrained model $\mathcal{M}_r$, which is trained exclusively on the remaining data $\mathcal{D}_r$.

To this end, our LTMU framework mainly consists of the following two steps (c.f. Figure 2): (1) Repairing of decision boundary, which repairs the decision boundary of the tail class using directional augmented features, guided by the similarity between the tail class and its neighboring classes; (2) Forgetting with boundary retraction, which retracts the repaired boundary away from the nearest remaining class, dispersing these features across the feature space to ensure comprehensive unlearning of the tail class.

3.1 Repairing of Decision Boundary

The first step is to repair the decision boundaries distorted by the limited sample size of tail classes, which aims to produce diverse features for the tail class, thereby expanding its feature space to facilitate complete removal in the subsequent unlearning phase. To analyze the relationship between the repair and unlearning in long-tailed setting, we primarily focus on the geometric effects of tail class unlearning through information augmentation techniques.

One significant distinction between long-tailed learning and unlearning methods lies in their application phases: long-tailed learning is applied during the training phase, while unlearning

Algorithm 1 Binary Search

Require: The original model $\mathcal{M}$, input feature z, target centroid $\mathbf{c}$, and threshold θ.
Ensure: Boundary feature $\bar{z}$.

Set $z_{target} \leftarrow \mathbf{c}$ and $z_{origin} \leftarrow z$;
while $||z_{target} - z_{origin}|| > \theta$ **do**
  Set $z_{mid} \leftarrow \frac{z_{target}+z_{origin}}{2}$;
  if $\mathcal{M}(z_{target}) = \mathcal{M}(z_{mid})$ **then**
    Set $z_{target} \leftarrow z_{mid}$;
  else
    Set $z_{origin} \leftarrow z_{mid}$;
  end if
end while
$\bar{z} \leftarrow z_{target}$;
return $\bar{z}$;

operates as a post-processing step. This difference underscores the importance of retaining information from the remaining data, $\mathcal{D}_r$, especially after repairing the distorted boundaries of tail classes in the unlearning process. Information augmentation techniques widely used in long-tailed learning can disrupt inter-class similarity [20], which may adversely impact the utility of neighboring classes. To address this issue, we propose a novel directional feature augmentation method that can preserve the inherent similarities between categories. Specifically, we generate compact augmented features that guide the original tail samples toward closer neighboring classes, while dispersing features toward more distant neighboring classes. This approach preserves essential inter-class relationships and maintains the integrity of the feature space.

Specifically, we first calculate the category centroids $\mathbf{c}$ for each class to capture the intrinsic inter-class similarity. To accurately obtain $\mathbf{c}$ in mini-batch data, we adopt the simple moving average algorithm [40]:

$$\begin{aligned} \mathbf{c}_j &= \frac{\sum z^j}{N_j}, \\ \mathbf{c}_j^l &= (1-\gamma)\mathbf{c}_j^l + \gamma \mathbf{c}_j^{(l-1)}, \end{aligned} \tag{1}$$

where $\mathbf{c}_j$ is the centroid of class j, N_j represents the number of samples in class j, l is the batch index and γ is the moving average coefficient. We denote the feature embedding of a sample $\boldsymbol{x}_i$ as $\boldsymbol{z}_i = E(\boldsymbol{x}_i)$, where E represents the feature extractor within a deep network. For each feature $\boldsymbol{z}_i$ in the tail class, we then identify the k closest centroids,

$$\mathbf{c}_k = \text{Top-}k\{\text{dist}(\mathbf{c}_j, \boldsymbol{z}_i \in \mathcal{D}_f) \mid j = 1, \ldots, C\}, \tag{2}$$

where $\text{dist}(\cdot)$ represents a similarity metric. In our scheme, Euclidean distance is sufficient for effective performance.

To guide the direction and control the compactness of the augmented features, we introduce the boundary features [23] positioned at the decision boundaries. Generating augmented features along the vectors extending from tail features to boundary features introduces greater representation diversity, helping to correct the distorted decision boundary of the tail class more effectively (c.f. Figure 2). Moreover, the distance between original features and their respective boundary features regulates the compactness of the

Algorithm 2 Augmented Feature Generation

Require: The original model $\mathcal{M}$, unlearning data $\mathcal{D}_f = (\boldsymbol{x}_i, \boldsymbol{y}_i)_{i=1}^m$, nearest class k, class centroids $\mathbf{c}$, feature extractor E and augmented number for each sample N.
Ensure: Generated augmented data $\mathcal{D}_{aug}$.

for $\boldsymbol{x}$ in $\mathcal{D}_f$ **do**
  Generate feature $\boldsymbol{z} = E(\boldsymbol{x})$;
  Select nearest k centroids $\mathbf{c}_k = \text{Top-}k\{\text{dist}(\mathbf{c}_j, \boldsymbol{z}) \mid j = 1, \ldots, C\}$;
  for i in $range(k)$ **do**
    Find boundary feature $\bar{\boldsymbol{z}} \leftarrow Binarysearch(\boldsymbol{z}, \mathbf{c}_i, \mathbf{w})$;
    for j in $range(N)$ **do**
      Calculate step $\boldsymbol{s} \leftarrow \frac{\bar{\boldsymbol{z}}-\boldsymbol{z}}{M}$;
      Generate augmented feature $\hat{\boldsymbol{z}} \leftarrow \boldsymbol{z} + j \times \boldsymbol{s}$;
      $\mathcal{D}_{aug} \leftarrow \mathcal{D}_{aug} \cup \{\hat{\boldsymbol{z}}, \boldsymbol{y}_{c_i}\}$;
    end for
  end for
end for
return $\bar{z}$;

augmented features, preserving inter-class similarity and thereby maintaining the utility of $\mathcal{D}_r$ throughout the unlearning process.

To locate the boundary features $\bar{\boldsymbol{z}}$, we use a simple binary search method (c.f. Algorithm 1). The step size for generating augmentation features is given by $\frac{\text{dist}(\bar{\boldsymbol{z}}_j - \boldsymbol{z}_i)}{M}$, where M is the total number of augmentation steps. The augmentation features $\hat{\boldsymbol{z}}$ can be then expressed as:

$$\hat{\boldsymbol{z}}_j = \boldsymbol{z}_i + m \times \frac{\text{dist}(\bar{\boldsymbol{z}}_j - \boldsymbol{z}_i)}{M}, m = 1, \ldots, M, \tag{3}$$

where m represents each incremental step taken in the direction towards the boundary, dividing the distance into M equal segments for generating augmented features.

So far, the augmented features for the tail class have been generated to repair its distorted decision boundary. The entire process is outlined in Algorithm 2. It is important to note that the repair process is carried out by finetuning the original model using both the original tail samples and the generated augmented features. Additionally, we combine this repair process with the subsequent unlearning process into a unified loss function.

3.2 Forgetting with Boundary Retraction

Following the generation of directional augmented features, we proceed to integrate them with the original features to achieve comprehensive forgetting of the targeted tail class. As revealed by a recent unlearning method, Boundary Shrink [6], the forgetting class can be removed by dispersing them across the feature space. This motivates us to approach our unlearning process from the perspective of decision boundary manipulation. We define class unlearning as the redistribution of the feature space of the forgetting class to the most similar neighboring classes. Therefore, we accomplish the forgetting of tail class by retracting the decision boundary between the tail class and its neighboring classes within the feature space.

However, this process faces two critical challenges: determining the placement of augmented features and setting the directions

for boundary retraction. Our directional repair scheme effectively addresses the positioning of augmented features, ensuring thorough unlearning of the tail class. Intuitively, samples from the tail class exhibit similarities to different neighboring classes, so we identify the k nearest neighboring classes for each tail sample, aiming to enhance the generalization capability of the tail class. For boundary retraction directions, unlike Boundary Shrink, which identifies neighboring classes using adversarially generated predictions, we leverage inter-class similarity based on Euclidean distance—computed during the repair phase—to guide boundary retraction in a structured and efficient way. The redistribution of the feature space is achieved by retracting the decision boundary of the tail class, which involves pushing samples across the decision boundary. The direction for boundary retraction for each tail and augmented feature is then determined by:

$$\boldsymbol{y}_i^{\text{nearest}} = \text{Top-1}\{\text{dist}(\mathbf{c}_j, \{\hat{\boldsymbol{z}}_i, E(\boldsymbol{x}_i)\} \in \mathcal{D}_f) \mid j = 1, \ldots, C\}, \quad (4)$$

where $\{\hat{\boldsymbol{z}}_i, E(\boldsymbol{x}_i)\}$ represents a set of features derived from the original tail sample $\boldsymbol{x}_i$ and the augmented features $\hat{\boldsymbol{z}}_i$. Here, E denotes the feature extractor of the model.

Thus, with a unified geometric interpretation of long-tailed learning and unlearning, we further enhance the efficiency of our unlearning by reusing the inter-class similarity. To retract the decision boundary of the forgetting tail class, we fine-tune the tail class samples toward the label of respective nearest neighboring class as:

$$\mathcal{L}_{unlearn} = \mathcal{L}(\{\hat{\boldsymbol{z}}_i, E(\boldsymbol{x}_i)\}, \boldsymbol{y}_i^{\text{nearest}}; \mathbf{w}_0), \quad (5)$$

where $\mathcal{L}$ denotes a standard loss function (e.g., cross-entropy in our case). Finally, this boundary retraction process can effectively ensure comprehensive forgetting of the tail class, while preserving the utility of the remaining classes.

4 Experiments

4.1 Experiment Setups

4.1.1 Datasets. We perform our experiments on two well-established long-tailed benchmarks: CIFAR10-LT and CIFAR100-LT. These benchmarks are derived from the balanced versions of CIFAR10 and CIFAR100 by applying an exponential decay function $n = n_i \rho^i$ [3], where i represents the class index (0-indexed), n_i is the original quantity of training images and ρ is the imbalanced factor defined by $\rho = N_{max}/N_{min}$. Both datasets are segmented into three distinct training datasets, each exhibiting varying levels of imbalance with factors of [100, 50, 10]. We primarily employ an imbalance factor of $\rho = 100$ to evaluate the efficacy of our unlearning approach under long-tailed conditions. Note that the original CIFAR10 and CIFAR100 datasets consist of 50,000 training images and 10,000 validation images, each with a resolution of 32×32 pixels. CIFAR10 contains 10 classes, while CIFAR100 comprises 100 classes.

We also adopt two widely used person re-identification datasets: MSMT17 [37] and DukeMTMC-ReID [43]. To study the difference of unlearning performance between head class and tail class, we construct two long-tailed datasets based on the original person ReID datasets. We classify the classes by ranking them according to the quantity of their samples. Specifically, the top 20 identities are designated as the head class, whereas the bottom 50 identities are considered as the tail classes. Consequently, the MSMT17 dataset presents an imbalance factor of 74.2, and the DukeMTMC-ReID dataset shows an imbalance factor of 71.0. Similarly, in the case of the CIFAR10-LT dataset, we pick the top 2 classes to be the head class and the last 2 classes to be the tail class. For the CIFAR100-LT dataset, the top 5 classes are selected as the head class and the last 10 classes are regarded as the tail class.

4.1.2 Evaluation Metrics. To evaluate the effectiveness of the unlearning methods, we use several metrics in line with the recent work [12], including unlearning accuracy (**UA**), unlearning test accuracy (**UTA**), remaining accuracy (**RA**), remaining test accuracy (**RTA**), attack success rate (**ASR**) of membership inference attack (MIA) on $\mathcal{D}_f$, and run-time efficiency (**RTE**). In particular, UA measures the accuracy of the unlearned model on $\mathcal{D}_f$, while UTA evaluates its accuracy on test data belonging to the forgetting class, denoted as $\mathcal{D}_{ft}$. RA assesses the model's performance on $\mathcal{D}_r$, indicating its ability to retain utility for $\mathcal{D}_r$. RTA, in contrast, evaluates the generalization capacity of the unlearned model on the remaining test data, denoted as $\mathcal{D}_{rt}$. Together, these accuracy metrics assess the utility preservation of the unlearning approach, ensuring that while the forgetting class utility is reduced, the performance on the remaining classes is retained. Moreover, ASR of MIA offers a measure of privacy guarantee, assessing whether membership in $\mathcal{D}_f$ can be detected through the unlearned model, which indicates the probability that a sample intended for forgetting is recognized as part of the training set. For an effective unlearning process, ASR should align closely with that of a model retrained without $\mathcal{D}_f$. Finally, RTE quantifies the time efficiency of the unlearning process, enabling comparison of the runtime performance across different unlearning methods.

4.1.3 Baselines. The following baselines are used for comparisons:
Retrain: Retraining from scratch without using the forgetting data, serving as the gold standard model that all unlearned models should approximate.
1) *Finetune* (**FT**) [36]: Finetuning the model on $\mathcal{D}_r$ for a few epochs to induce catastrophic forgetting, producing the unlearned model.
2) *Random Labels* (**RL**) [15]: Finetuning the model by randomly re-labeling samples from $\mathcal{D}_f$ to disrupt the influence of $\mathcal{D}_f$ on the model.
3) *Gradient Ascent* (**GA**) [32]: Training the model on $\mathcal{D}_f$ using gradient ascent to reverse the learned information.
4) *L1-Sparse* (**L1**) [24]: Finetuning the model on $\mathcal{D}_r$ while applying an l_1 regularizer to eliminate information associated with $\mathcal{D}_f$.
5) *Bad Teacher* (**BT**) [7]: Using selective knowledge distillation to transfer only the information from D_r into the unlearned model.
6) *Boundary Shrink* (**BS**) [6]: Shrinking the feature space of $\mathcal{D}_f$ by moving the decision boundaries towards the nearest neighboring class.
7) *SalUn* (**SU**) [12]: Removing parameters highly influenced by $\mathcal{D}_f$ through hard thresholding based on saliency maps.

4.1.4 Implementations. We use PyTorch for training on GeForce RTX 4090 GPUs. We adopt the ResNet-18 (resp. ResNet-34) architecture for CIFAR10-LT (resp. CIFAR100-LT), and both models undergo 200 training epochs with Stochastic Gradient Descent (SGD), employing a momentum of 0.9, a weight decay of 5e-3, and a cosine

Table 1: Performance of different unlearning methods on CIFAR10-LT and MSMT17 across two unlearning scenarios. The results are given by $a(\pm b)$, where a denotes the mean value, b denotes the standard deviation over 5 independent trails. The most outstanding unlearning performance for each method is emphasized with bold text, while the second-best is distinguished with an underline.

Dataset	Method	Head class					Tail class				
		UA	UTA	RA	RTA	ASR	UA	UTA	RA	RTA	ASR
ResNet-18 on CIFAR10-LT	Retrain	0.00	0.00	100.00	77.12	0.47	0.00	0.00	99.55	79.63	0.66
	FT	**0.00**	**0.00**	96.75(± 0.03)	76.71(±1.55)	0.20(±0.02)	44.20(±1.19)	10.45(±1.65)	98.61(±0.58)	71.11(±4.32)	0.38(±0.05)
	RL	**0.00**	**0.00**	91.92(±0.15)	71.44(±1.52)	0.29(±0.05)	33.82(±4.18)	5.40(±2.30)	99.39(±0.25)	75.90(±0.94)	0.34(±0.04)
	GA	5.45(±3.19)	3.95(±2.45)	91.86(±0.65)	72.14(±4.46)	0.38(±0.03)	33.22(±4.79)	5.40(±2.30)	98.91(±0.21)	75.92(±0.95)	0.34(±0.04)
	L1	7.22(±1.10)	7.65(±3.25)	95.45(±0.50)	78.57(±1.14)	0.05(±0.03)	**0.00**	**0.00**	99.78(±0.11)	75.39(±0.46)	0.69(±0.05)
	BT	**0.00**	**0.00**	92.15(±0.92)	70.95(±0.32)	0.34(±0.07)	46.66(±5.34)	25.15(±1.15)	98.79(±0.87)	74.43(±2.33)	0.06(±0.03)
	BS	**0.00**	**0.00**	91.91(±3.93)	64.37(±1.17)	0.50(±0.08)	39.84(±6.16)	12.28(±3.75)	**99.85(±0.35)**	79.39(±1.10)	0.60(±0.10)
	SU	0.06(±0.01)	0.83(±0.15)	**97.12(±0.23)**	75.49(±1.31)	**0.49(±0.05)**	8.58(±2.01)	3.90(±0.74)	99.17(±0.28)	81.52(±1.47)	0.64(±0.06)
	Ours	0.06(±0.01)	0.05(±0.01)	92.28(±0.29)	**79.39(±1.41)**	0.51(±0.09)	**0.00**	0.10(±0.05)	99.46(±0.45)	**81.78(±2.14)**	**0.68(±0.09)**
ResNet-50-IBN on MSMT17	Retrain	0.00	0.00	99.56	95.53	0.51	0.00	0.00	99.48	95.52	0.74
	FT	**0.00**	**0.00**	97.96(±0.39)	91.27(±0.76)	0.74(±0.04)	**0.00**	**0.00**	98.05(±1.24)	90.83(±0.87)	0.91(±0.08)
	RL	1.27(±0.73)	0.64(±0.09)	97.17(±1.42)	94.70(±1.92)	0.55(±0.03)	41.35(±3.18)	70.97(±4.76)	99.18(±1.08)	93.44(±1.19)	0.81(±0.11)
	GA	0.90(±0.08)	1.11(±0.10)	97.68(±1.39)	95.18(±0.89)	0.63(±0.02)	53.33(±2.11)	68.24(±1.90)	99.19(±0.35)	93.53(±0.48)	0.38(±0.11)
	L1	**0.00**	**0.00**	96.01(±1.88)	92.95(±2.01)	0.51(±0.05)	**0.00**	**0.00**	96.54(±0.96)	91.90(±1.04)	0.39(±0.06)
	BT	**0.00**	1.61(±1.22)	**98.66(±1.14)**	**96.45(±1.96)**	0.55(±0.06)	**0.00**	**0.00**	98.24(±1.46)	95.05(±1.02)	0.38(±0.06)
	BS	0.42(±0.21)	0.18(±0.03)	97.30(±1.36)	94.53(±1.94)	0.51(±0.07)	66.32(±6.67)	75.89(±4.01)	99.53(±0.08)	94.70(±0.14)	0.63(±0.02)
	SU	1.99(±0.54)	0.85(±0.21)	98.32(±1.04)	95.31(±1.52)	0.50(±0.10)	6.21(±1.50)	4.33(±0.81)	95.12(±1.23)	91.96(±1.23)	0.76(±0.05)
	Ours	**0.00**	**0.00**	98.16(±0.36)	95.79(±0.17)	**0.51(±0.01)**	**0.00**	**0.00**	**99.62(±0.81)**	**95.67(±0.66)**	**0.74(±0.03)**

annealing learning rate scheduler. The batch size is set to 64. Following [26], we utilize a pre-trained ResNet-50-IBN for MSMT17 and DukeMTMC-reID datasets, fine-tuning this model for an additional 40 epochs to optimize the performance. For our LTMU framework, only the parameters specific to the unlearning stage need to be configured. The learning rate is uniformly set to 5e-4 for 10 epochs across all datasets. The parameter k, representing the number of most similar categories, is dataset-dependent: $k = 4$ for CIFAR10-LT, $k = 16$ for CIFAR100-LT, MSMT17, and DukeMTMC-ReID. Additionally, the total number of augmentation steps, M, is fixed at 5 for all datasets, as variations in M have been found to exert minimal influence on the performance of the unlearning method.

For training the baseline models, we employ consistent parameters with those used in training the original model for the Retrain baseline, applied to the remaining dataset $\mathcal{D}_r$. For the **FT** baseline, the original DNN model is fine-tuned on $\mathcal{D}_r$ using a higher learning rate of 1.5e-3 over 10 epochs. For the **RL** baseline, the original model is fine-tuned on the randomly relabeled dataset $\mathcal{D}_f$, with a learning rate of 1e-4 maintained for 10 epochs. In the **GA** baseline, the original model is fine-tuned on $\mathcal{D}_f$ using gradient ascent, which maximizes the loss function on $\mathcal{D}_f$. This fine-tuning process is conducted with a learning rate of 1e-5 for 10 epochs.

For the **L1** baseline, the original model is fine-tuned with an additional l_1 regularizer. Following the setup in [24], the sparsity-promoting regularization parameter is set to 5e-3, which decays linearly over 10 epochs. The learning rate for this fine-tuning process is set to 5e-4.

In the **BT** baseline, an incompetent teacher is utilized to remove information pertaining to $\mathcal{D}_f$, while a competent teacher preserves information from $\mathcal{D}_r$. Knowledge distillation is then employed to transfer knowledge from both teachers to the unlearned model. For this approach, the learning rate is set to 1e-4, and the training is conducted over 10 epochs.

The **BS** baseline leverages adversarial attacks to identify the nearest incorrect label. Specifically, the Fast Gradient Sign Method (FGSM) is used for this purpose. The hyper-parameters are configured as follows: the attack bound is set to 1.0, the learning rate is 5e-4, and the training process spans 10 epochs.

For the **SU** baseline, we select the top-20% most salient parameters based on the absolute gradient of the forgetting data and train only these using the *Finetune* pipeline. Training is performed for 10 epochs with a cosine scheduler and learning rate of 3e-4.

4.2 Unlearning Performance

We first report the experimental results of our LTMU with six baseline methods on CIFAR10-LT and MSMT17 across the long-tailed setting in Table 1 (additional results for CIFAR100-LT and DukeMTMC-ReID are provided in Appendix B.1). For successful unlearning, UA and UTA would ideally remain low, while RA and RTA are expected to stay relatively high. Retrain serves as the upper performance bound for unlearning approaches. The results reveal that most baselines achieve some degree of unlearning when forgetting data belongs to the head class; for instance, FT, RL, BT and BS maintain low UA and UTA, reflecting that the utility of $\mathcal{D}_f$ is removed in these models. However, when tasked with unlearning a tail class, most baselines struggle to erase information from $\mathcal{D}_f$, as evidenced by higher UA and UTA values. This challenge likely arises due to distorted decision boundaries surrounding tail classes, limiting the success of standard unlearning. Leveraging directional feature augmentation, our LTMU framework effectively forgets $\mathcal{D}_f$ information even in tail classes, as demonstrated by consistently low UA and UTA.

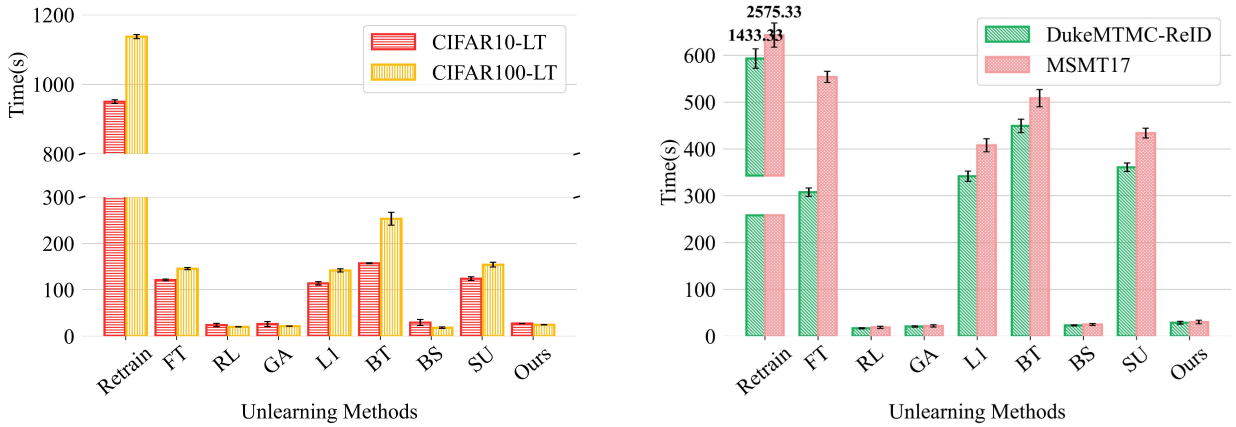


Figure 3: The time consumption of each unlearning method on four datasets.

Regarding privacy, our LTMU achieves ASR results closely matching those of Retrain across datasets, highlighting its robustness against membership inference attacks. Furthermore, LTMU shows minimal performance gaps compared to the Retrained model across both head and tail unlearning scenarios, achieving notably low UA for $\mathcal{D}_f$ at only 2.14% on CIFAR10-LT and 0.00% on MSMT17, while maintaining high RTA on $\mathcal{D}_r$. ASR results affirm that LTMU provides privacy resilience superior to other baselines across most evaluations.

Our LTMU framework performs comparably to the Retrain model on most metrics and shows unique advantages on certain evaluation criteria. The L1 approach achieves similar results to Retrain on the CIFAR10-LT dataset, but it requires extensive fine-tuning on all $\mathcal{D}_r$, resulting in higher unlearning time (detailed in Section 4.3). Additionally, the L1 approach does not consistently maintain performance across all datasets. Furthermore, LTMU not only achieves lower UA and UTA values but also demonstrates a higher RTA than the Retrain model. This indicates that our directional feature augmentation scheme effectively enhances the generalization ability of the unlearned model on the remaining classes.

4.3 Unlearning Efficiency

Beyond utility and privacy guarantees, minimizing time consumption is essential when model owners receive forgetting requests in real-world applications. So here we evaluate the unlearning efficiency by measuring the time each method requires to complete unlearning. As shown in Figure 3, LTMU significantly outperforms other unlearning baselines. Specifically, LTMU achieves substantial speed-ups over Retrain, with 34.82× and 46.11× faster on CIFAR10-LT and CIFAR100-LT, respectively. On ReID datasets, where Retrain incurs high time costs, LTMU's acceleration is even more pronounced, highlighting its effectiveness in scenarios with high computational demands. While LTMU requires additional time to generate augmented features, it compensates by utilizing class similarity instead of adversarial labels, which aligns its time consumption with BS. Additionally, while some unlearning methods (L1 and BT) yield satisfactory performance in utility and privacy guarantees, they are too time-consuming to be practical. This is largely due to their coarse unlearning approaches, which necessitate extensive fine-tuning on remaining data to recover utility.

4.4 Ablation Studies

To demonstrate the improvements introduced by our directional feature augmentation scheme, we compare unlearning performance using different repair schemes, implemented via random Input Augmentation (IA) [31], Feature Augmentation (FA) [41], and our Directional Feature Augmentation (DFA) methods. The detailed implementations of IA and FA are provided in Appendix A.2. We present the unlearning performance of two baseline methods (GA and BS), using IA and FA augmentation schemes, alongside our framework in Table 2. The results show that using input-level augmentations like IA for tail class unlearning retains residual information from $\mathcal{D}_f$ and significantly reduces the utility of $\mathcal{D}_r$. This is likely due to IA repairing the decision boundary of the tail class to a certain extent while failing to preserve inter-class similarity. In contrast, FA generally preserves the utility of $\mathcal{D}_r$, but it reduces the generalization capability of the unlearned model on $\mathcal{D}_{rt}$ (lower RTA). This suggests that while FA can more effectively repair the decision boundary of the tail class, it still causes blurring of the decision boundary for the remaining classes. Overall, our LTMU framework with DFA repair achieves the best unlearning performance. By reusing inter-class similarity and seamlessly integrating it into the framework, LTMU maintains a satisfactory RTE score, despite DFA being a more complex repair scheme. Note that integrating other unlearning baselines (FT, RL, L1 and BT) with FA is challenging, as they either do not use $\mathcal{D}_f$ or do not compute the feature space. Thus, a direct comparison requires separately executing the repair and unlearning tasks. The conclusions remain consistent with those above, and we also provide these additional results in Table 2. Lastly, our unified boundary-based framework is significantly more time-efficient compared to these separate schemes.

4.5 Hyper-parameter Analysis

The top k similar classes is critical in our LTMU framework, as it determines the number and orientation of generated features for the tail class. So next we analyze the impact of varying k on performance metrics for all four datasets, as shown in Figure 4. Our

Table 2: Comparison of using Input Augmentation (IA), Feature Augmentation (FA) and our method on Cifar10-LT and MSMT17 on tail class. The results are given by $a(\pm b)$, sharing the same format with Table 1

Dataset	Method	UA	UTA	RA	RTA	ASR	RTE
ResNet-18 on CIFAR10-LT	Retrain	0.00	0.00	99.55	79.63	0.66	950.31
	FT+IA	28.05(±2.18)	9.69(±1.12)	98.41(±0.80)	76.57(±1.02)	0.42(±0.03)	133.30(±5.78)
	FT+FA	7.23(±1.58)	9.04(±0.96)	99.45(±0.23)	78.44(±0.83)	0.45(±0.08)	146.90(±9.06)
	RL+IA	26.51(±2.84)	31.20(±2.72)	96.97(±0.71)	74.71(±1.52)	0.69(±0.07)	34.02(±0.98)
	RL+FA	0.45(±0.09)	0.40(±0.03)	98.83(±0.51)	75.94(±2.53)	0.71(±0.05)	40.81(±1.17)
	GA+IA	3.61(±1.58)	7.30(±0.94)	92.22(±1.38)	72.21(±1.56)	0.73(±0.07)	24.87(±1.69)
	GA+FA	0.61(±0.15)	0.40(±0.04)	98.67(±0.47)	72.13(±1.97)	**0.68(±0.03)**	**22.16(±0.92)**
	BS+IA	3.79(±0.75)	4.00(±0.49)	65.73(±2.14)	37.13(±3.08)	0.77(±0.06)	29.57(±1.69)
	BS+FA	7.83(±2.66)	9.60(±1.87)	99.41(±0.53)	74.88(±1.54)	0.40(±0.14)	26.74(±1.82)
	L1+IA	**0.00**	**0.00**	99.68(±0.09)	74.88(±0.64)	0.81(±0.11)	137.56(±4.77)
	L1+FA	**0.00**	**0.00**	**99.89(±0.08)**	75.56(±0.87)	0.78(±0.07)	140.27(±5.10)
	BT+IA	13.54(±1.89)	7.50(±0.30)	93.82(±2.58)	75.73(±0.63)	0.35(±0.10)	171.23(±6.01)
	BT+FA	2.10(±0.33)	2.90(±0.28)	99.09(±0.14)	75.62(±0.88)	0.44(±0.01)	188.09(±10.59)
	Ours	**0.00**	**0.10(±0.05)**	99.46(±0.45)	**81.78(±2.14)**	0.68(±0.09)	27.29(±1.09)
ResNet-50-IBN on MSMT17	Retrain	0.00	0.00	99.48	95.52	0.74	2575.33
	FT+IA	31.33(±3.51)	39.67(±2.09)	98.22(±0.35)	88.14(±0.98)	0.92(±0.07)	591.08(±11.62)
	FT+FA	2.00(±0.50)	0.40(±0.02)	98.77(±0.19)	89.48(±1.10)	0.85(±0.10)	607.22(±17.48)
	RL+IA	22.58(±1.55)	42.31(±3.19)	94.38(±0.87)	80.69(±1.11)	0.87(±0.04)	36.76(±2.41)
	RL+FA	6.15(±0.39)	11.22(±2.03)	96.54(±0.09)	82.11(±1.03)	0.80(±0.09)	39.35(±3.88)
	GA+IA	26.67(±4.16)	24.49(±2.15)	98.58(±0.47)	91.35(±1.47)	0.66(±0.08)	24.51(±1.32)
	GA+FA	**0.00**	**0.00**	98.02(±1.30)	87.57(±0.93)	0.57(±0.05)	**21.83(±1.79)**
	BS+IA	15.25(±1.37)	24.49(±2.48)	98.45(±0.41)	91.11(±1.40)	0.52(±0.03)	29.01(±2.11)
	BS+FA	**0.00**	12.58(±1.06)	98.49(±0.38)	93.60(±1.38)	0.50(±0.04)	25.88(±1.33)
	L1+IA	**0.00**	**0.00**	95.19(±1.03)	79.15(±3.56)	0.10(±0.02)	446.31(±12.75)
	L1+FA	**0.00**	**0.00**	97.23(±0.76)	89.27(±0.91)	0.33(±0.10)	507.59(±10.10)
	BT+IA	11.28(±0.29)	6.93(±0.15)	92.82(±1.08)	91.89(±1.83)	0.50(±0.04)	523.47(±9.98)
	BT+FA	**0.00**	**0.00**	97.09±0.14	90.31(±0.99)	0.46(±0.06)	601.94(±20.05)
	Ours	**0.00**	**0.00**	**99.62(±0.81)**	**95.67(±0.66)**	**0.74(±0.03)**	30.54(±1.39)

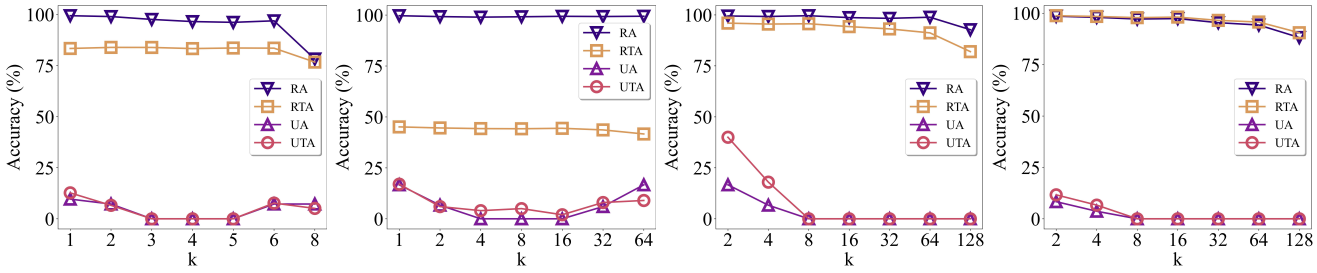


Figure 4: Utility performance of the LTMU framework on CIFAR10-LT, CIFAR100-LT, MSMT17 and DukeMTMC-reID datasets (left to right) for different values of top k similar classes used in directional feature augmentation.

findings reveal that, with an increasing k, UA and UTA initially decrease and then increase, while RA and RTA show a slight overall decline. These trends suggest that choosing either too few or too many nearest classes can reduce the effectiveness of our method. Furthermore, using an excessively large k during unlearning may degrade RA and RTA, ultimately impacting the utility of the unlearned model. Our results indicate that selecting a moderate value for k yields stable and effective unlearning performance according to the size of label space across different datasets. That is why we set k to 4 and 16 for CIFAR10-LT and CIFAR100-LT, and $k = 16$ for both MSMT17 and DukeMTMC-reID.

In addition, the number of augmentation steps, denoted as M, in our boundary repair scheme is a key hyperparameter in our framework. It controls the number of augmented samples generated in each direction. To evaluate the impact of M, we assess the utility performance of our LTMU with varying values of M. The results in Figure 5 indicate that with a small number of steps ($M = 1, 3$),

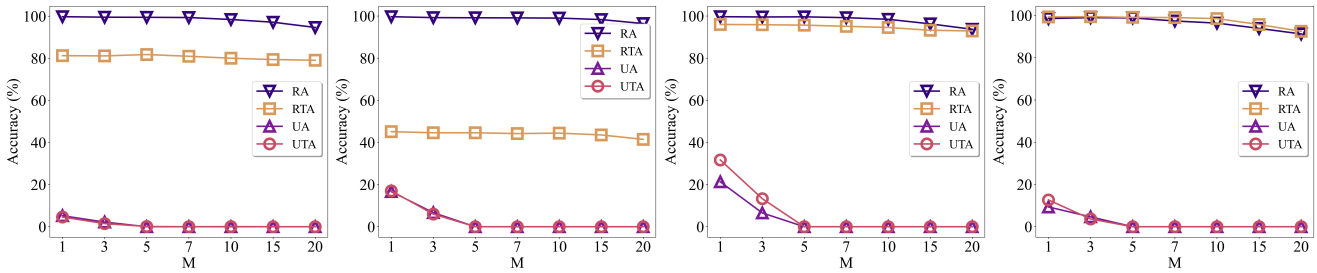


Figure 5: Utility performance of the LTMU framework on CIFAR10-LT, CIFAR100-LT, MSMT17 and DukeMTMC-reID datasets (left to right) for varying number of augmentation steps M used in directional feature augmentation.

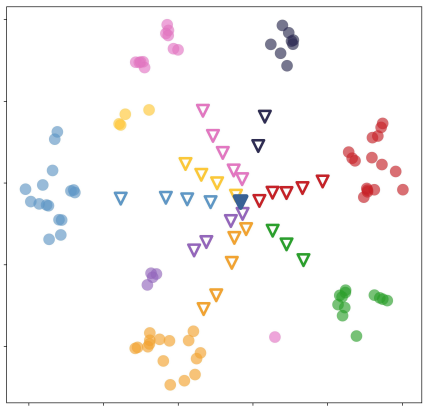
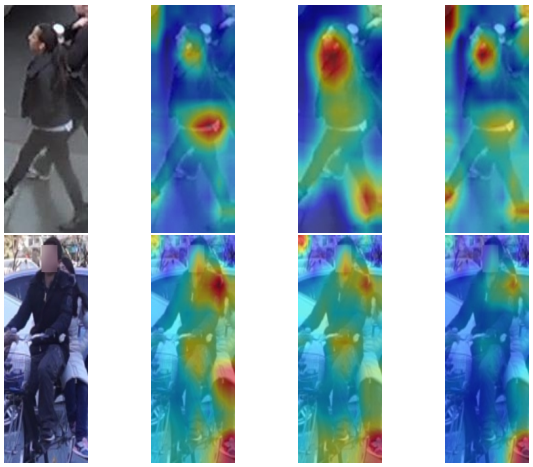


Figure 6: t-SNE visualization of features on MSMT17 dataset. The central blue solid triangle represents the feature of a tail class sample, while the colored solid dots correspond to features of head class samples. The colored hollow triangles indicate generated features that are directed towards different head classes, guided by inter-class similarity.



(a) Input **(b) Original** **(c) Retrain** **(d) Unlearn**

Figure 7: The attention maps for the original, retrained, and unlearned models, demonstrating the areas of focus on the unlearning data from the ReID dataset.

both UA and UTA remain relatively high across all datasets, suggesting that fewer augmentation samples retain some information about $\mathcal{D}_f$. As M increases, LTMU demonstrates superior and stable unlearning performance for the tail class.

4.6 Visualization of Tail Class

To clarify the workings of our directional feature augmentation, we present in Figure 6 a t-SNE [33] visualization of both the original features from the tail and head classes (represented by solid triangles and dots, respectively) and the augmented features (represented by hollow triangles). The augmented features are directed towards the centroids of their top k similar classes and are labeled to their respective class accordingly. During the unlearning process, the generated features guide the retracting of decision boundaries. This organized boundary shifting enables an effective splitting of the repaired feature space for the tail class, facilitating the removal of information associated with $\mathcal{D}_f$.

To transparently verify the effect of our LTMU, we further generate the attention maps for models on unlearning data from the ReID dataset, as illustrated in Figure 7. The highlighted regions represent areas of model attention. Observably, the unlearned model produced by our LTMU shifts its focus to the corners and the background of person images, indicating its effective removal of specific tail identification information.

5 Conclusion

In this paper, we have proposed a novel machine unlearning approach for long-tailed data distributions, focusing on the challenge of effectively unlearning tail classes. Through a tailored bidirectional boundary manipulation, our approach can fully erase the tail class while maintaining the utility of the remaining classes. In particular, we introduce a directional boundary repair scheme operating in the feature space to enhance the representational capability of tail classes. Building on this repair scheme, we further develop a boundary retraction approach to disperse the tail class features. Experiments across multiple datasets and neural network architectures validate the effectiveness and superiority of our approach. Our work provides a practical solution for long-tailed machine unlearning and paves the way for future research on unlearning methods tailored to real-world, imbalanced data distributions.

References

[1] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. 2021. Machine unlearning. In *Proceedings of IEEE S&P*. 141–159.

[2] Jonathan Brophy and Daniel Lowd. 2021. Machine unlearning for random forests. In *Proceedings of ICML*. 1092–1104.

[3] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. 2019. Learning imbalanced datasets with label-distribution-aware margin loss. In *Proceedings of NeurIPS*.

[4] Yinzhi Cao and Junfeng Yang. 2015. Towards making systems forget with machine unlearning. In *Proceedings of IEEE S&P*. 463–480.

[5] Huiqiang Chen, Tianqing Zhu, Xin Yu, and Wanlei Zhou. 2024. Machine Unlearning via Null Space Calibration. *CoRR, arXiv: 2404.13588* (2024).

[6] Min Chen, Weizhuo Gao, Gaoyang Liu, Kai Peng, and Chen Wang. 2023. Boundary unlearning: Rapid forgetting of deep networks via shifting the decision boundary. In *Proceedings of IEEE/CVF CVPR*. 7766–7775.

[7] Vikram S Chundawat, Ayush K Tarun, Murari Mandal, and Mohan Kankanhalli. 2023. Can bad teaching induce forgetting? unlearning in deep networks using an incompetent teacher. In *Proceedings of AAAI*. 7210–7217.

[8] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. 2020. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of IEEE/CVF CVPR Workshops*. 702–703.

[9] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. 2019. Class-balanced loss based on effective number of samples. In *Proceedings of IEEE/CVF CVPR*. 9268–9277.

[10] Lydia de la Torre. 2018. A guide to the california consumer privacy act of 2018. *Available at SSRN 3275571* (2018).

[11] Fei Du, Peng Yang, Qi Jia, Fengtao Nan, Xiaoting Chen, and Yun Yang. 2023. Global and local mixture consistency cumulative learning for long-tailed visual recognitions. In *Proceedings of IEEE/CVF CVPR*. 15814–15823.

[12] Chongyu Fan, Jiancheng Liu, Yihua Zhang, Eric Wong, Dennis Wei, and Sijia Liu. 2024. SalUn: Empowering Machine Unlearning via Gradient-based Weight Saliency in Both Image Classification and Generation. In *Proceedings of ICLR*.

[13] Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. 2015. Certifying and removing disparate impact. In *proceedings of ACM SIGKDD*. 259–268.

[14] Antonio Ginart, Melody Guan, Gregory Valiant, and James Y Zou. 2019. Making ai forget you: Data deletion in machine learning. In *Proceedings of NeurIPS*.

[15] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. 2020. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of IEEE/CVF CVPR*. 9304–9312.

[16] Laura Graves, Vineel Nagisetty, and Vijay Ganesh. 2021. Amnesiac machine learning. In *Proceedings of AAAI*. 11516–11524.

[17] Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. 2020. Certified data removal from machine learning models. In *Proceedings of ICML*. 3832–3842.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of IEEE/CVF CVPR*. 770–778.

[19] Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. 2019. Deep imbalanced learning for face recognition and attribute prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42, 11 (2019), 2781–2794.

[20] Tao Huang, Shan You, Fei Wang, Chen Qian, and Chang Xu. 2022. Knowledge distillation from a stronger teacher. In *Proceedings of NeurIPS*. 33716–33727.

[21] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. *Master's Thesis* (2009).

[22] Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. 2024. Towards unbounded machine unlearning. In *Proceedings of NeurIPS*.

[23] Gaoyang Liu, Zehao Tian, Jian Chen, Chen Wang, and Jiangchuan Liu. 2023. TEAR: Exploring temporal evolution of adversarial robustness for membership inference attacks against federated learning. *IEEE Transactions on Information Forensics and Security* 18 (2023), 4996–5010.

[24] Jiancheng Liu, Parikshit Ram, Yuguang Yao, Gaowen Liu, Yang Liu, PRANAY SHARMA, Sijia Liu, et al. 2024. Model sparsity can simplify machine unlearning. In *Proceedings of NeurIPS*.

[25] Yang Liu, Mingyuan Fan, Cen Chen, Ximeng Liu, Zhuo Ma, Li Wang, and Jianfeng Ma. 2022. Backdoor defense with machine unlearning. In *Proceedings of IEEE INFOCOM*. 280–289.

[26] Hao Luo, Youzhi Gu, Xingyu Liao, Shenqi Lai, and Wei Jiang. 2019. Bag of tricks and a strong baseline for deep person re-identification. In *Proceedings of IEEE/CVF CVPR Workshops*.

[27] Yanbiao Ma, Licheng Jiao, Fang Liu, Maoji Wen, Lingling Li, Wenping Ma, Shuyuan Yang, Xu Liu, and Puhua Chen. 2025. Predicting and Enhancing the Fairness of DNNs with the Curvature of Perceptual Manifolds. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2025).

[28] Yanbiao Ma, Licheng Jiao, Fang Liu, Shuyuan Yang, Xu Liu, and Puhua Chen. 2023. Feature distribution representation learning based on knowledge transfer for long-tailed classification. *IEEE Transactions on Multimedia* 26 (2023), 2772–2784.

[29] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. 2019. Robustness via curvature regularization, and vice versa. In *Proceedings of IEEE/CVF CVPR*. 9078–9086.

[30] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. 2016. Performance Measures and a Data Set for Multi-Target, Multi-Camera Tracking. In *Proceedings of ECCV Workshops*. 17–35.

[31] Connor Shorten and Taghi M Khoshgoftaar. 2019. A survey on image data augmentation for deep learning. *Journal of Big Data* 6, 1 (2019), 1–48.

[32] Anvith Thudi, Gabriel Deza, Varun Chandrasekaran, and Nicolas Papernot. 2022. Unrolling sgd: Understanding factors influencing machine unlearning. In *Proceedings of IEEE EuroS&P*. 303–319.

[33] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9 (2008), 2579–2605.

[34] Paul Voigt and Axel Von dem Bussche. 2017. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing* 10, 3152676 (2017), 10–5555.

[35] Jianfeng Wang, Thomas Lukasiewicz, Xiaolin Hu, Jianfei Cai, and Zhenghua Xu. 2021. Rsg: A simple but effective module for learning imbalanced datasets. In *Proceedings of IEEE/CVF CVPR*. 3784–3793.

[36] Alexander Warnecke, Lukas Pirch, Christian Wressnegger, and Konrad Rieck. 2021. Machine unlearning of features and labels. *CoRR, arXiv:2108.11577* (2021).

[37] Longhui Wei, Shiliang Zhang, Wen Gao, and Qi Tian. 2018. Person Transfer GAN to Bridge Domain Gap for Person Re-Identification. In *Proceedings of IEEE/CVF CVPR*. 79–88.

[38] Miao Xu. 2024. Machine Unlearning: Challenges in Data Quality and Access. In *Proceedings of IJCAI*.

[39] Xi Yin, Xiang Yu, Kihyuk Sohn, Xiaoming Liu, and Manmohan Chandraker. 2019. Feature transfer learning for face recognition with under-represented data. In *Proceedings of IEEE/CVF CVPR*. 5704–5713.

[40] Chenyu You, Yifei Mint, Weicheng Dai, Jasjeet S Sekhon, Lawrence Staib, and James S Duncan. 2024. Calibrating multi-modal representations: A pursuit of group robustness without annotations. In *Proceedings IEEE/CVF CVPR*. 26140–26150.

[41] Yuhang Zang, Chen Huang, and Chen Change Loy. 2021. Fasa: Feature augmentation and sampling adaptation for long-tailed instance segmentation. In *Proceedings of IEEE/CVF ICCV*. 3457–3466.

[42] Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. 2015. Scalable Person Re-Identification: A Benchmark. In *Proceedings of IEEE/CVF ICCV*. 1116–1124.

[43] Zhedong Zheng, Liang Zheng, and Yi Yang. 2017. Unlabeled Samples Generated by GAN Improve the Person Re-Identification Baseline in Vitro. In *Proceedings of IEEE/CVF ICCV*. 3754–3762.

Appendices

The appendix includes four parts. The first part describes the detailed experiment settings (Sec. A). The second part presents additional experimental results with respect to unlearning results on diverse datasets (Sec. B.1), impact of various augmentation methods (Sec. B.2) and impact of different imbalance factors (Sec. B.3). The last part discusses the limitations of the proposed framework (Sec. C).

A Additional Information of Experiments

A.1 Descriptions of ReID Datasets

To evaluate the effectiveness of the proposed LTMU approach in real-world scenarios and at scale, we conduct experiments on three widely used person re-identification datasets: DukeMTMC-ReID, MSMT17, and Market-1501.

- The Duke-MTMC-ReID dataset [43] is a subset of the multi-target multi-camera tracking (MTMCT) dataset DukeMTMC [30], collected using eight static HD cameras across Duke University's campus.
- MSMT17 [37] is a large-scale ReID benchmark captured in outdoor and indoor environments using 15 cameras, offering significant variation in lighting and background conditions.

- Market-1501 [42] was collected at Tsinghua University using six cameras (five high-resolution and one low-resolution), covering the area outside a campus supermarket.

All datasets contain pedestrian bounding box images resized to a uniform resolution of 256×128 pixels. These benchmarks provide diverse and challenging conditions for evaluating the unlearning performance of our method.

A.2 Implementation Details of Input Augmentation and Feature Augmentation

Information augmentation is a widely adopted strategy to enhance the quantity and diversity of imbalanced datasets [31]. Building upon this foundation, we extend the concept of information augmentation to the unlearning scenario. By leveraging information augmentation techniques, we potentially address challenges inherent to imbalanced data in some extent, improving the performance of the unlearning method. This approach enables the model to better adapt and recalibrate its learned knowledge, particularly for underrepresented tail classes. In our framework, we employ two typical information augmentation techniques: Input Augmentation (IA) [8] and Feature Augmentation (FA) [41].

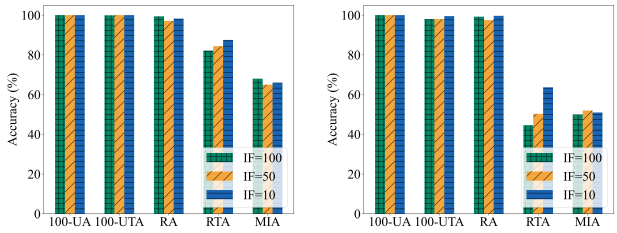
Input Augmentation (IA): This technique manipulates input data directly to create diverse variations. Following the methodology in [8], we utilize a two-step process. First, we increase the quantity of tail class data to 128 by duplicating the forgetting data. Next, we apply a set of randomly selected transformations to these images, including flip, color adjustment, brightness adjustment, sharpness enhancement, and equalization. The resulting augmented dataset, denoted as $\mathcal{D}_{f,\text{aug}}$, is then used in the unlearning process.

Feature Augmentation (FA): Unlike input augmentation, feature augmentation operates within the learned feature space. Following in [41], we begin by extracting features $\boldsymbol{z}_i = E(\boldsymbol{x}_i)$ from all input data in the forgetting data. Next, we compute two statistical metrics: the mean feature vector $\bar{z}$ and the covariance matrix $\Sigma_{\boldsymbol{z}}$. Using these, we sample noise $\xi \sim \mathcal{N}(0, \Sigma_{\boldsymbol{z}})$ from the covariance matrix. The sampled noise is then added to the mean feature vector, yielding augmented features $\hat{\boldsymbol{z}} = \bar{z} + \xi$. This process is repeated until the number of augmented features reaches 64.

B Extensive Experiments

B.1 Comparative Analysis of Performance Metrics Across Diverse Datasets

To further verify the effectiveness of our unlearning method, we report the experimental results of our LTMU with six baseline methods on extensive datasets CIFAR100-LT, DukeMTMC-ReID and Market-1501 in Table 3. The results reveal that our LTMU method is competitive with the Retrain model across the majority of metrics, while also demonstrating distinct advantages in specific evaluation criteria. For instance, in the DukeMTMC-ReID dataset, our LTMU method secures the top performance in terms of UA, UTA and RTA. Additionally, it ranks second in both RA and ASR across both head class and tail class scenarios. This aligns with the findings presented in the main text, further validating the effectiveness of our approach.



(a) CIFAR10-LT (b) CIFAR100-LT

Figure 8: The performance of our method on CIFAR10-LT and CIFAR100-LT with different imbalance factor.

B.2 Impact of Augmentation Methods

To evaluate the potential risks associated with input augmentation (IA) in blurring decision boundaries for the remaining classes, we conducted experiments on four unlearning baselines, comparing their performance with and without IA. The results are summarized in Table 4. The findings reveal that the use of IA results in a reduction in both UA and UTA for the baseline models. However, it is accompanied by a noticeable decline in RA and RTA, as well as a significant increase in ASR. These results suggest that IA not only compromises the integrity of decision boundaries for the remaining classes but also introduces substantial privacy risks, underscoring the need for careful consideration when employing IA in long-tailed unlearning scenarios.

B.3 Impact of Different Imbalance Factors

To evaluate the robustness of our LTMU method, we conducted experiments under varying imbalance factors ρ, specifically at $\rho = 100$, $\rho = 50$, and $\rho = 10$. The results are depicted in Figure 8. As shown, our performance metrics remain highly consistent across these different imbalance scenarios. Notably, the RTA improves as the imbalance factor decreases, which can be attributed to the reduced skewness in the data distribution, thereby enhancing the model's generalizability. These findings collectively demonstrate the robustness of our LTMU method across a wide range of imbalance factors.

C Discussion of Limitations

LTMU addresses several critical challenges in machine unlearning, particularly within the domain of image classification. This study primarily centers on applications in Computer Vision, emphasizing its fundamental importance to the unlearning problem. However, the generalization of the proposed techniques to other domains, such as Natural Language Processing (NLP), remains an open question. The transferability and efficacy of these methods in NLP or other non-vision domains require further comprehensive investigation, highlighting a key limitation and potential direction for future research. Besides, LTMU makes significant strides in advancing the field of class-wise unlearning. However, the problem of sample-wise unlearning, where the forgetting process involves samples from multiple classes, has not been addressed in this study. We acknowledge these limitations and plan to explore these directions in future work.

Table 3: Performance of different unlearning methods on CIFAR100-LT, DukeMTMC-ReID and Market-1501 across two unlearning scenarios. The results are given by $a(\pm b)$, where a denotes the mean value, b denotes the standard deviation over 5 independent trails. The most outstanding unlearning performance for each method is emphasized with bold text, while the second-best is distinguished with an underline.

Dataset	Method	Head class					Tail class				
		UA	UTA	RA	RTA	ASR	UA	UTA	RA	RTA	ASR
ResNet-18 on CIFAR100-LT	Retrain	0.00	0.00	80.43	46.29	0.48	0.00	0.00	99.76	46.87	0.52
	FT	6.81(±0.76)	5.80(±0.27)	**94.36(±0.84)**	44.96(±0.85)	0.20(±0.05)	32.18(±2.01)	11.50(±1.50)	**99.79(±0.10)**	43.18(±0.21)	0.67(±0.05)
	RL	11.12(±2.43)	4.40(±1.96)	76.85(±1.96)	40.99(±2.13)	0.37(±0.04)	15.53(±0.66)	9.00(±1.22)	99.19(±0.15)	42.44(±1.02)	0.33(±0.08)
	GA	8.48(±2.03)	2.60(±0.80)	76.97(±1.78)	41.08(±2.06)	0.37(±0.04)	12.11(±2.89)	6.55(±1.25)	95.34(±1.92)	38.43(±2.03)	0.42(±0.08)
	L1	24.69(±4.32)	15.60(±3.61)	88.61(±0.30)	45.06(±1.34)	0.08(±0.03)	16.67(±2.39)	8.83(±0.36)	99.19(±0.04)	42.45±0.11)	0.33(±0.08)
	BT	**0.00**	**0.00**	79.71(±0.28)	40.51(±0.74)	0.21(±0.07)	**0.00**	**0.00**	97.27(±0.05)	40.22(±0.78)	0.10(±0.03)
	BS	2.80(±1.04)	0.60(±0.08)	76.93(±3.93)	41.54(±1.39)	**0.51(±0.02)**	15.17(±1.22)	4.30(±0.70)	95.27(±0.20)	38.51(±2.05)	0.60(±0.07)
	Ours	0.80(±0.07)	**0.00**	80.22(±0.41)	**47.24(±1.40)**	0.53(±0.08)	**0.00**	**0.00**	99.16(±0.04)	**44.58(±0.11)**	**0.50(±0.09)**
ResNet-50-IBN on DukeMTMC-ReID	Retrain	0.00	0.00	99.56	95.53	0.51	0.00	0.00	99.48	95.52	0.55
	FT	**0.00**	**0.00**	97.96(±1.05)	91.27(±0.96)	0.84(±0.11)	**0.00**	**0.00**	98.05(±0.48)	90.83(±0.89)	0.84(±0.04)
	RL	1.31(±0.85)	1.09(±0.43)	97.66(±1.44)	98.75(±0.75)	0.41(±0.06)	70.33(±5.24)	36.67(±3.08)	98.60(±0.51)	98.98(±0.27)	0.33(±0.02)
	GA	1.82(±0.11)	1.57(±0.30)	97.48(±1.07)	98.80(±0.63)	0.38(±0.07)	73.67(±4.88)	53.84(±6.01)	97.51(±0.79)	96.01(±1.27)	0.69(±0.07)
	L1	**0.00**	**0.00**	96.01(±0.39)	92.95(±1.28)	**0.51(±0.02)**	**0.00**	7.69(±2.44)	96.54(±1.47)	91.90(±2.20)	0.66(±0.07)
	BT	**0.00**	**0.00**	**98.66(±1.01)**	96.45(±0.89)	0.55(±0.10)	**0.00**	**0.00**	98.43(±0.44)	95.34(±0.96)	**0.58(±0.08)**
	BS	0.42(±0.27)	0.18(±0.05)	97.30(±0.63)	94.53(±1.17)	0.51(±0.03)	43.38(±2.55)	61.47(±1.69)	**99.53(±0.15)**	94.70(±0.08)	0.63(±0.03)
	Ours	**0.00**	**0.00**	98.59(±0.71)	**98.86(±0.59)**	**0.51(±0.02)**	**0.00**	**0.00**	98.85(±1.06)	**99.12(±0.13)**	0.60(±0.03)
ResNet-50-IBN on Market-1501	Retrain	0.00	0.00	99.56	95.53	0.51	0.00	0.00	99.48	95.52	0.74
	FT	**0.00**	**0.00**	97.96(±0.93)	91.27(±1.22)	0.71(±0.06)	**0.00**	**0.00**	98.05(±1.05)	90.83(±2.45)	0.84(±0.09)
	RL	1.27(±0.25)	0.64(±0.18)	97.27(±0.22)	94.69(±0.36)	0.41(±0.06)	71.67(±4.21)	84.16(±2.60)	97.53(±0.85)	94.61(±0.90)	0.37(±0.11)
	GA	0.90(±0.39)	1.11(±0.27)	97.68(±0.55)	95.18(±0.21)	0.66(±0.05)	65.33(±1.34)	87.54(±3.59)	98.55(±1.01)	94.74(±2.18)	0.42(±0.05)
	L1	**0.00**	**0.00**	96.01(±3.00)	92.95(±2.17)	0.51(±0.07)	**0.00**	**0.00**	96.54(±1.11)	91.90(±2.39)	**0.67(±0.12)**
	BT	**0.00**	**0.00**	**98.66(±0.32)**	**96.45(±0.52)**	0.55(±0.02)	**0.00**	**0.00**	96.43(±1.22)	93.34(±2.70)	0.56(±0.06)
	BS	0.42(±0.08)	0.18(±0.02)	97.30(±1.18)	94.53(±0.60)	0.51(±0.01)	61.37(±2.11)	70.29(±3.91)	**99.53(±0.15)**	94.70(±1.25)	0.63(±0.03)
	Ours	**0.00**	**0.00**	96.43(±1.19)	96.32(±1.23)	**0.50(±0.03)**	**0.00**	**0.00**	98.36(±1.08)	**94.95(±0.90)**	0.66(±0.04)

Table 4: Comparison of w/wo using Input Augmentation(IA) on tail class of Cifar10-LT and Cifar100-LT. The results are given by $a(\pm b)$, sharing the same format with Table 3.

Dataset	Method	UA	UTA	RA	RTA	ASR	RTE
ResNet-18 on CIFAR10-LT	Retrain	0.00	0.00	99.55	79.63	0.66	950.31
	RL	33.82(±4.18)	**5.40(±2.30)**	99.39(±0.25)	75.90(±0.94)	0.34(±0.04)	**24.02(±0.56)**
	GA	**33.22(±4.79)**	**5.40(±2.30)**	98.91(±0.21)	75.92(±0.95)	0.34(±0.04)	26.13(±1.78)
	BT	46.66(±5.34)	25.15(±1.15)	98.79(±0.87)	74.43(±2.33)	0.06(±0.03)	157.23(±11.15)
	BS	39.84(±6.16)	12.28(±3.75)	**99.85(±0.35)**	**79.39(±1.10)**	**0.60(±0.10)**	25.43(±1.32)
	RL+IA	**3.41(±0.52)**	4.25(±0.88)	93.95(±1.24)	73.04(±0.83)	0.74(±0.10)	**26.94(±1.17)**
	GA+IA	3.41(±0.87)	**4.15(±0.62)**	93.96(±0.69)	73.08(±1.08)	**0.73(±0.06)**	29.87(±1.69)
	BT+IA	31.09(±4.41)	15.3(±0.89)	92.35(±1.00)	**74.97(±1.91)**	0.25(±0.08)	219.86(±5.11)
	BS+IA	4.08(±0.87)	4.90(±1.22)	**94.13(±1.14)**	73.83(±0.92)	0.75(±0.06)	29.57(±1.69)
ResNet-34 on CIFAR100-LT	Retrain	0.00	0.00	99.76	46.87	0.52	1137.51
	RL	15.53(±0.66)	9.00(±1.22)	**99.19(±0.15)**	**42.44(±1.02)**	0.33(±0.08)	**20.15(±0.25)**
	GA	12.11(±4.88)	6.55(±1.25)	95.34(±1.92)	38.43(±2.03)	0.42(±0.07)	21.65(±0.28)
	BT	**0.00**	**0.00**	97.27(±0.44)	40.22(±0.96)	0.10(±0.03)	253.29(±13.97)
	BS	15.17(±1.22)	4.30(±0.70)	95.27(±0.20)	38.51(±2.05)	**0.60(±0.03)**	21.19(±1.73)
	RL+IA	**0.00**	2.04(±0.84)	55.91(±3.28)	30.55(±4.11)	0.83(±0.09)	24.87(±1.36)
	GA+IA	**0.00**	32.11(±2.15)	55.86(±0.47)	30.54(±1.47)	0.85(±0.08)	**24.51(±1.32)**
	BT+IA	**0.00**	**0.00**	**79.03(±1.81)**	**41.86(±2.53)**	0.00	293.87(±9.21)
	BS+IA	3.79(±0.19)	4.00(±0.33)	65.73(±2.42)	37.13(±3.06)	**0.77(±0.10)**	24.79(±0.86)