REMAINING-DATA-FREE MACHINE UNLEARNING BY SUPPRESSING SAMPLE CONTRIBUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Machine unlearning (MU) aims to remove the influence of specific training samples from a well-trained model, a task of growing importance due to the "right to be forgotten." The unlearned model should approach the retrained model, where forgetting data do not contribute to the training process. Therefore, unlearning should withdraw their contribution from the pre-trained model. However, quantifying and disentangling sample's contribution to overall learning process is highly challenging, leading most existing MU approaches to adopt other heuristic strategies such as random labeling or knowledge distillation. These operations inevitably degrade model utility, requiring additional maintenance with remaining data. To advance MU towards better utility and efficiency for practical deployment, we seek to approximate sample contribution with only the pre-trained model. We theoretically and empirically reveal that sample's contribution during training manifests in the learned model's increased sensitivity to it. In light of this, we propose MU-Mis (Machine Unlearning by Minimizing input sensitivity), which directly suppresses the contribution of forgetting data. This straightforward suppression enables MU-Mis to successfully unlearn without degrading model utility on the remaining data, thereby eliminating the need for access to the remaining data. To the best of our knowledge, this is the first time that a remaining-data-free method can outperform state-of-the-art (SOTA) unlearning methods that utilize the remaining data.

1 INTRODUCTION

Deep neural networks (DNNs) are revealed to store information of training data (Feldman, 2020; Feldman & Zhang, 2020; Tian et al., 2025) and such information could be reproduced by privacy attacks (Shokri et al., 2017; Zhu et al., 2019), raising data privacy concerns. The "right to be forgotten" (Regulation, 2018) is introduced to safeguard user privacy, which entails ensuring that the DNN performs as if the data were never involved in the training.

While retraining from scratch would ideally achieve this, it is often infeasible due to the high cost of training DNNs. This has motivated the study of *"Machine Unlearning"* (MU) (Cao & Yang, 2015), which fine-tunes the *pre-trained model* to approach the *retrained model* as closely as possible. The essential distinction in pre-trained and retrained model lies in the contribution of forgetting data, whose role shift from "contributors" that affect parameter updates in the pre-trained model to "bystanders" that exert no influence in the retrained model. Therefore, unlearning should aim to withdraw their contribution to the learning process.

However, identifying such a contribution is highly challenging. Learning is a dynamic process that gradually remembers and assimilates data, while unlearning, which is the reverse process that gradually removes data information, is achieved by backtracking the training trajectory to withdraw historical gradients in early study (Graves et al., 2021; Thudi et al., 2022). Nevertheless, such tracking not only contradicts the efficiency demands of unlearning but also yields limited effectiveness due to the incrementality of training (Wang et al., 2024b).

Consequently, most existing MU methods circumvent the difficulty of estimating sample contribution through other heuristics. A common strategy is to introduce confusion, *e.g.*, random relabeling (Golatkar et al., 2020; Graves et al., 2021; Fan et al., 2024) or knowledge distillation from useless teacher (Chundawat et al., 2023; Kurmanji et al., 2024). However, these approaches suffer from

several limitations: *(i)* such confusion causes *catastrophe unlearning* (Wang et al., 2024b) or *over-forgetting* (He et al., 2025), *i.e.*, severe degradation of model utility on the remaining data; *(ii)* the degradation in turn necessitates costly maintenance using the remaining data, thereby substantially *undermining MU efficiency*; *(iii)* the remaining data are not always accessible in practice. These limitations collectively underscore the importance of moving beyond heuristic confusion strategies and developing more principled unlearning mechanisms to advance MU toward higher utility and efficiency. Therefore, although quantifying sample contribution is inherently challenging, in this paper, we make efforts to ground unlearning in a precise characterization of sample's contribution.

Instead of accumulating the historical contributed gradient update during training, we identify the clue of contribution directly from the derivative of the training algorithm w.r.t a training sample. The learning process is a mapping by the training algorithm $\mathcal{A}$ from the training set $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}$ to a learned function f: denoted as $f = \mathcal{A}(\mathcal{D})$. Therefore, the training sample $\boldsymbol{x}_i$ contributes to the output: $\partial\mathcal{A}/\partial\boldsymbol{x_i} \neq 0$ while a sample out of the training set does not. A simple yet enlightening example lies in the support vector machine (Cortes & Vapnik, 1995; Christmann & Steinwart, 2008), where only the training data can act as support vectors that impact the decision boundary. Thus, withdrawing the sample contribution can be achieved by suppressing $\partial\mathcal{A}/\partial\boldsymbol{x}_i$.

The main challenge is that $\mathcal{A}$ corresponds to a dynamic training process without a closed-form expression. To address this, we theoretically illustrate that $\partial\mathcal{A}/\partial\boldsymbol{x}_i$ could be approximated by the learned model's sensitivity to its input $\boldsymbol{x}$, i.e. $\partial f(\boldsymbol{x})/\partial\boldsymbol{x}$ with $f = \mathcal{A}(\mathcal{D})$ in *Section 3.2*. To derive a principled and optimization-friendly guideline aligned with the behavior of a retrained model, we delve deeper into the input sensitivity across different logits. Our empirical investigations under the machine learning (*Section 3.3*) and machine unlearning (*Section 3.4*) scenarios reveal that a sample's contribution manifests as disproportionately higher input sensitivity of the target logit relative to irrelevant logits. In light of this finding, we propose **MU-Mis** (Machine Unlearning by Minimizing Input Sensitivity), which suppresses sample contribution by reducing the sensitivity disparity between the target and non-target logits to the forgetting data.

We evaluate MU-Mis on 3 standard unlearning tasks across 6 datasets, benchmarking against 6 competitive remaining-data-dependent unlearning methods and 4 existing remaining-data-free baselines. The results demonstrate that MU-Mis achieves effective unlearning while preserving model utility on the remaining data **without utilizing them**, performing on par with SoTA remaining-data-dependent approaches and outperforming all remaining-data-free methods significantly, with the added advantage of notable computational efficiency. Moreover, due to its principled forgetting mechanism, MU-Mis exhibits stable and effective behavior in sequential unlearning, whereas existing methods are disclosed to exhibit several deficiencies. Collectively, these results underscore the practicality and reliability of MU-Mis for real-world deployment.

Our key contributions can be summarized as follows:

❶ We theoretically and empirically reveal that a sample's contribution is reflected in the amplified sensitivity gap between the target logit and irrelevant logits, enabling the identification of sample contribution with only the pre-trained model.

❷ Based on the above analysis and findings, we propose MU-Mis, which suppresses the sample's contribution by minimizing the sensitivity magnitude gap for the forgetting data.

❸ Comprehensive experiments demonstrate the effectiveness and efficiency of MU-Mis. To our best knowledge, it is the first time that a remaining-data-free method can outperform SoTA remaining-data-dependent methods.

2 Related Work

In Appendix D, we provide a detailed review of existing machine unlearning research, which can be broadly categorized into three lines: *(i) gradient-based* methods trace sample contributions through stored training information but are costly and fragile in DNNs; *(ii) loss-guided re-optimization* methods erase target data through knwoledge confusion or parameter suppression but inevitably hurts model utility on the remaining data and require access to the remaining data; *(iii) remaining-data-free methods* avoid reliance on the remaining data but still suffer from poor utility or auxiliary data requirements. While existing literatures provide valuable insights, they either fall short in terms

of utility or practicality. In contrast, we introduce a lightweight and optimizable approximation of sample contributions that avoids backtracking learning trajectory and minimizes utility degradation, thereby eliminating additional maintenance with the remaining data. This balance of effectiveness and efficiency makes our approach more practical for real-world deployment than existing methods.

3 Machine Learning, Machine Unlearning and Input Sensitivity

3.1 Problem Formulation

Machine Learning (ML) is to learn a mapping from the input space $\mathcal{X}$ to the output space $\mathcal{Y}$, denoted as the function $f(\cdot) : \mathcal{X} \to \mathcal{Y}$. As we mainly focus on classification models, the output of f is C-dimensional in a C-category classification model. Learning is performed by a *training algorithm* $\mathcal{A}$, which generally takes in a training dataset $\mathcal{D}$ and returns the *learned function* f, *i.e.*, the outcome of $\mathcal{A}$ varies with different training datasets. To investigate sample-wise influence on the learning process, we consider $\mathcal{A}$ in a broader sense and distinguish different training processes by the *training dataset* $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^m$. That is to say, we have a family of the training algorithm $\mathcal{A}_\mathcal{D}$ and each one is a multivariate function that takes all the samples $\{\boldsymbol{x}_j \in \mathcal{X}\}$ as input, regardless of whether they are in the training dataset $\mathcal{D}$. Therefore, the output of $\mathcal{A}_\mathcal{D}$ does not vary with each input variable, but only varies with the change of the training data $\boldsymbol{x}_i \in \mathcal{D}$, and makes no response to the change of samples out of the training set.

Machine Unlearning (MU) is to remove the influence of *forgetting data* $\mathcal{D}_f \subset \mathcal{D}$ from the *pre-trained model* w_p, while preserving model utility on the *remaining data* $\mathcal{D}_r = \mathcal{D} \backslash \mathcal{D}_f$. The learned function f is parameterized by parameters $w \in \mathbb{R}^d$ with input variable $\boldsymbol{x}$, *i.e.*, instantiated as $f(\boldsymbol{x}; w)$. A good approximate unlearning mechanism should efficiently and effectively transform w_p into a *sanitized model* w_u, such that the output distribution of w_u closely matches *retrained model* w_r.

Remark on notation. To facilitate the understanding of the objectives in our analysis, we only bold the input variables of the training algorithm $\mathcal{A}_\mathcal{D}$ and learned function $f(\boldsymbol{x})$, which are respectively $\boldsymbol{x_i}$ and $\boldsymbol{x}$ in the following analysis.

3.2 Theoretical analysis connecting sample contribution and input sensitivity

As previously discussed, machine unlearning is to withdraw sample's contribution to the learning process, and an efficient unlearning method should explore the contribution directly from the pre-trained model. To detach per-sample contribution with the pre-trained model, we propose to identify the clue of contribution from the derivative of training mapping $\mathcal{A}_\mathcal{D}$ to training sample $\boldsymbol{x_i}$, i.e. $\partial \mathcal{A}_\mathcal{D} / \partial \boldsymbol{x_i}$. Recall that $\mathcal{A}_\mathcal{D}$ is determined by the training dataset $\mathcal{D} = \{(\boldsymbol{x_i}, \boldsymbol{y}_i)\}_{i=1}^m$ and outputs the learned function $f(\boldsymbol{x}; w_p)$. Then $\partial \mathcal{A}_\mathcal{D} / \partial \boldsymbol{x_i}$ is to compute $\partial f(\boldsymbol{x}; w_p) / \partial \boldsymbol{x_i}$. However, there is no explicit expression for this derivative. Therefore, in this part, we reflect on the learning dynamics to seek a surrogate with the pre-trained model. Figure 1 provides an overview of the key objectives investigated in our following analysis.

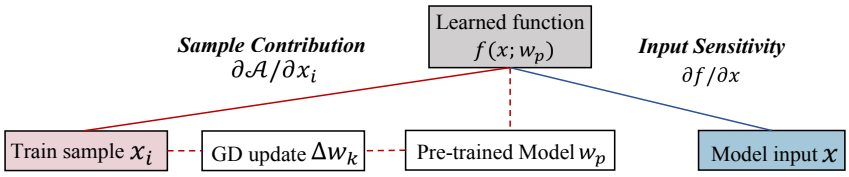

Figure 1: **A brief overview of the theoretical connection between sample's contribution and a pre-trained model's input sensitivity.** The dashed arrows illustrate how the influence of a training sample propagates through gradient updates to the pre-trained model.

Gradient Descent (GD). After T iterations training updates in the parameter space, we have pre-trained model parameter $w_p = w_0 + \sum_{k=1}^T \Delta w_k$, where w_0 is randomly initialized model parameters and $\Delta w_k = w_{k+1} - w_k$ is the k^{th} parameter update. Specifically, when training loss $\mathcal{L}$ and gradient descent with step size η are used, we have

$$\Delta w_k = -\eta \sum_{i=1}^m \frac{\partial \mathcal{L}(\boldsymbol{x_i})}{\partial w}\Big|_{w=w_k} = -\eta \sum_{i=1}^m \frac{\partial f(\boldsymbol{x_i}; w)}{\partial w}\Big|_{w=w_k} \frac{\partial \mathcal{L}(\boldsymbol{x_i})}{\partial f}.$$

Function space update induced by GD. Viewing machine learning from the function space with first-order Taylor expansion on parameters, correspondingly we have $f = f_0 + \sum_{k=1}^{T} \Delta f_k$, where $f_0 = f(\boldsymbol{x}, w_0)$ is initial function and Δf_k is induced by parameter update Δw_k. The evolution in function induced by parameter update is:

$$\Delta f_k(\boldsymbol{x}; w) \approx \frac{\partial f(\boldsymbol{x}; w)}{\partial w}\Big|_{w=w_k}^{\top} \Delta w_k = -\eta \sum_{i=1}^{m} \frac{\partial f(\boldsymbol{x}; w)}{\partial w}\Big|_{w=w_k}^{\top} \frac{\partial f(\boldsymbol{x_i}; w)}{\partial w}\Big|_{w=w_k} \frac{\partial \mathcal{L}(\boldsymbol{x_i})}{\partial f}.$$

Learned function. To better explain the idea, we make simplifications: *(i)* Note that $\frac{\partial f(\cdot; w)}{\partial w}\big|_{w=w_k}$ is the mapping from model input $\boldsymbol{x}$ to the induced backpropagation gradient with parameters w_k. We abbreviate this mapping as $g_k(\boldsymbol{x}) : \mathcal{X} \to \mathbb{R}^{d \times C}$ and its derivative to input $\boldsymbol{x}$ as g_k', where d is total number of model parameters. *(ii)* In classification problem with cross-entropy loss as $\mathcal{L}$, we have $\frac{\partial \mathcal{L}(\boldsymbol{x_i})}{\partial f} = e_c - p(\boldsymbol{x_i})$, where e_c is a one-hot vector with only c^{th} element equals to 1, and p is the probability vector of $\boldsymbol{x_i}$. The final learned function f is

$$f(\boldsymbol{x}; w_p) = f(\boldsymbol{x}; w_0) + \sum_{k=1}^{T} \Delta f_k(\boldsymbol{x}, w) = f(\boldsymbol{x}; w_0) - \eta \sum_{k=1}^{T} \underbrace{g_k^{\top}(\boldsymbol{x})}_{(1)} \sum_{i=1}^{m} \underbrace{g_k(\boldsymbol{x_i})(e_c - p(\boldsymbol{x_i}))}_{(2)}.$$

Notice that term (1) is related to the **forward inference process** while term (2) is related to the **machine learning process**. Derivative of f w.r.t $\boldsymbol{x}$ indicates how the prediction of f varies with its input $\boldsymbol{x}$ at inference time, while derivative w.r.t $\boldsymbol{x_i}$ indicates how the learned function f varies when the training sample $\boldsymbol{x_i}$ varies. The former implies **the learned model's sensitivity to its input**, and the latter is **the training sample's influence on learning**. Next, we take the derivative of f w.r.t $\boldsymbol{x}$ and $\boldsymbol{x_i}$ respectively to view their relationship. Note that p is a probability vector determined by $\boldsymbol{x}_i$. Due to softmax activation, we consider $p(\boldsymbol{x_i})$ hardly changes around $\boldsymbol{x_i}$, and omit its derivative term w.r.t $\boldsymbol{x_i}$. The difference in mapping g_k when $\boldsymbol{x_i}$ changes is also omitted.

$$\begin{cases} \dfrac{\partial f(\boldsymbol{x}; w_p)}{\partial \boldsymbol{x_i}} = -\eta \sum_{k=1}^{T} \underbrace{g_k'(\boldsymbol{x_i}) g_k(\boldsymbol{x})(e_c - p(\boldsymbol{x_i}))}_{=:\mathcal{C}_k(\boldsymbol{x}, \boldsymbol{x_i})}, \\ \dfrac{\partial f(\boldsymbol{x}; w_p)}{\partial \boldsymbol{x}} = \dfrac{\partial f(\boldsymbol{x}; w_0)}{\partial \boldsymbol{x}} - \eta \sum_{k=1}^{T} \sum_{i=1}^{m} \underbrace{g_k'(\boldsymbol{x}) g_k(\boldsymbol{x_i})(e_c - p(\boldsymbol{x_i}))}_{=:\mathcal{S}_k(\boldsymbol{x}, \boldsymbol{x_i})}. \end{cases} \tag{1}$$

Input sensitivity of learned function reflects sample contribution. $\mathcal{C}_k(\boldsymbol{x}, \boldsymbol{x_i})$ determines the prediction change on $\boldsymbol{x}$ when $\boldsymbol{x_i}$ changes, and $\mathcal{S}_k(\boldsymbol{x}, \boldsymbol{x_i})$ stands for the part of model's sensitivity to $\boldsymbol{x}$ contributed by training sample $\boldsymbol{x_i}$. Note that $\frac{\partial f(\boldsymbol{x_i}; w_p)}{\partial \boldsymbol{x_i}}$ is similar to the definition of memorization, which is framed as *self-influence* (Feldman, 2020; Feldman & Zhang, 2020). To be more specific, memorization of a sample is defined as the prediction difference in itself when training with or without it. Similarly, the self-influence here is the prediction difference on $\boldsymbol{x_i}$ when it slightly changes, i.e. $\frac{\partial f(\boldsymbol{x_i}; w_p)}{\partial \boldsymbol{x_i}}$. Thus we consider $\frac{\partial f(\boldsymbol{x_i}; w_p)}{\partial \boldsymbol{x_i}}$ as the reflection of sample $\boldsymbol{x_i}$'s contribution. From the formulation, we have $\mathcal{S}_k(\boldsymbol{x_i}, \boldsymbol{x_i}) = \mathcal{C}_k(\boldsymbol{x_i}, \boldsymbol{x_i})$. For a specific training sample $\hat{x} \in \mathcal{D}$, the learned model's sensitivity to it can be further decomposed as

$$\begin{aligned} \frac{\partial f(\boldsymbol{x}; w_p)}{\partial \boldsymbol{x}}\Big|_{\boldsymbol{x}=\hat{x}} &= \frac{\partial f(\boldsymbol{x}; w_0)}{\partial \boldsymbol{x}}\Big|_{\boldsymbol{x}=\hat{x}} - \eta \sum_{k=1}^{T} \sum_{i=1}^{m} \mathcal{S}_k(\hat{x}, \boldsymbol{x_i}) \\ &= \frac{\partial f(\boldsymbol{x}, w_0)}{\partial \boldsymbol{x}}\Big|_{\boldsymbol{x}=\hat{x}} - \eta \sum_{k=1}^{T} \left[\mathcal{S}_k(\hat{x}, \hat{x}) + \sum_{\tilde{x} \in \mathcal{D}/\hat{x}} \mathcal{S}_k(\hat{x}, \tilde{x}) \right] \\ &= \underbrace{-\eta \sum_{k=1}^{T} \mathcal{S}_k(\hat{x}, \hat{x})}_{\text{Contribution Term}} + \underbrace{\frac{\partial f(\boldsymbol{x}, w_0)}{\partial \boldsymbol{x}}\Big|_{x=\hat{x}} - \eta \sum_{k=1}^{T} \sum_{\tilde{x} \in \mathcal{D}/\hat{x}} \mathcal{S}_k(\hat{x}, \tilde{x})}_{\text{Residual Term}}. \end{aligned} \tag{2}$$

The randomly initialized function f_0 is generally quite insensitive to input change. Thus, the first term of the above residual term is very small. The second term is related to the correlation between the gradient on $\tilde{x}$ and the sensitivity of the gradient on $\hat{x}$. We use a simple MLP model to illustrate the insight of $\mathcal{S}_k(\hat{x}, \tilde{x}) << \mathcal{S}_k(\hat{x}, \hat{x})$ with $\hat{x} \neq \tilde{x}$ in Appendix E. Therefore, the residual term is relatively smaller than the contribution term. In summary, the contribution of a training sample to the training process would be approximately reflected in the pre-trained model's output sensitivity to the sample.

Empirical validation. We validate the contribution to learning by comparing $\|\nabla_{\boldsymbol{x}} f\|_F$ of the training data before and after training in Figure 2. In a randomly initialized model, there is little response to input changes, only about 10^{-4}. After training, there is a significant order of magnitude growth to 10^3, indicating an increased attention of the trained model to the training data's variations. This implies that the training data contribute to model performance, and such efforts include promoting the model's sensitivity to them during training.

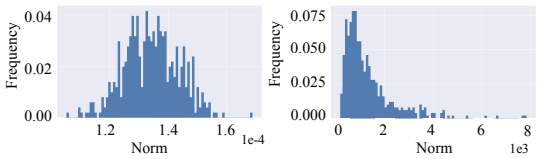

Figure 2: **Input sensitivity $\|\nabla_x f\|_F$ of training data before and after training.** Left: In randomly initialized model w_0. Right: In well-trained model w_p. After training, the model exhibits significantly increased sensitivity to the training data, reflecting their contribution during training.

3.3 Input sensitivity of the target and irrelevant class logit

During training, model predictions on samples are driven toward their correct labels, so sample contributions might differ across logits. To further refine our view of sample contribution, we examine individual logits of $f(\boldsymbol{x}) \in \mathbb{R}^C$ in the following part.

Let f_c denotes the logit output of the target class and $f_{c'}$ denotes the logit output of irrelevant classes. Figure 3 compares distributions between $\|\nabla_{\boldsymbol{x}} f_c\|_F$ and $\frac{1}{C-1}\sum_{c'\neq c}\|\nabla_{\boldsymbol{x}} f_{c'}\|_F$ (denoted as $\|\nabla_{\boldsymbol{x}} f_{c'}\|_F$ for brevity in the following) of training data before and after training. In the **randomly initialized** model, these two quantities are of **comparative magnitude**, but $\|\nabla_{\boldsymbol{x}} f_c\|_F$ becomes **much larger** than $\|\nabla_{\boldsymbol{x}} f_{c'}\|_F$ **after training**. This observation implies that samples contribute to amplifying $\|\nabla_{\boldsymbol{x}} f_c\|_F$ to surpass $\|\nabla_{\boldsymbol{x}} f_{c'}\|_F$ during training, generating a discernible difference in whether a sample has been learned. A complementary explanation of this finding comes from the generative view of discriminative models: the softmax-based discriminative classifier is revealed to be implicitly a density model which learns data distribution (Grathwohl et al.; Srinivas & Fleuret, 2021). From this viewpoint, the logits $f(\boldsymbol{x})$ of standard classifiers are un-normalized log-densities, and corresponding input-gradients $\nabla_{\boldsymbol{x}} f_i(\boldsymbol{x})$ are log-gradients of a class-conditional density model. In other words, we have $\nabla_{\boldsymbol{x}} \log p_\theta(\boldsymbol{x}|y=i) = \nabla_{\boldsymbol{x}} f_i(\boldsymbol{x})$ in the classification model, providing a rationale for the observed discrepancy.

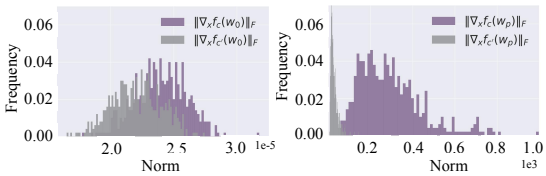

Figure 3: **Input sensitivity $\|\nabla_x f_c\|_F$ and $\|\nabla_x f_{c'}\|_F$ before and after training.** Left: randomly initialized model w_0. Right: well-trained model w_p. After training, the gap between target and irrelevant class sensitivities enlarges, providing a clearer signal of the sample's contribution.

3.4 Input sensitivity of samples present and absent in training

For effective unlearning, the optimization objective should accurately steer the pre-trained model toward the retrained model. To validate that the theoretically grounded sensitivity gap provides a reliable measure of sample contribution to guide unlearning, we empirically examine the input sensitivity of forgetting data under MU scenarios (introduced in Section 5.1 and Appendix G.1).

For each forgetting sample, we compute the difference Δ between the retrained and the pre-trained model's sensitivity to it, where the sensitivity including $\|\nabla_{\boldsymbol{x}} f_c\|_F$, $\|\nabla_{\boldsymbol{x}} f_{c'}\|_F$ and $\|\nabla_{\boldsymbol{x}} f_c\|_F - \|\nabla_{\boldsymbol{x}} f_{c'}\|_F$. Aiming for a light-weight unlearning algorithm, we prefer an optimization direction rather than modeling a distribution or specifying a target value for each sample. Hence, we focus on the *sign* of Δ and count the ratio of rise and fall of Δ to examine the overall trend in Figure 4.

From left to right in Figure 4, Δ is the sample-wise difference between the retrained and pre-trained model on $\|\nabla_{\boldsymbol{x}} f_c\|_F$, $\|\nabla_{\boldsymbol{x}} f_{c'}\|_F$ and $\|\nabla_{\boldsymbol{x}} f_c\|_F - \|\nabla_{\boldsymbol{x}} f_{c'}\|_F$. For each quantity, there is a consistent trend across different unlearning settings. Generally, f_c of the retrained model exhibits lower sensitivity and $f_{c'}$ exhibits higher sensitivity to the forgetting data than the pre-trained model. And their sensitivity magnitude gap is consistently smaller in the retrained model across different settings. Therefore, the sensitivity magnitude gap faithfully reflects the behavior of the retrained model and thus serves as a reliable objective to guide unlearning.

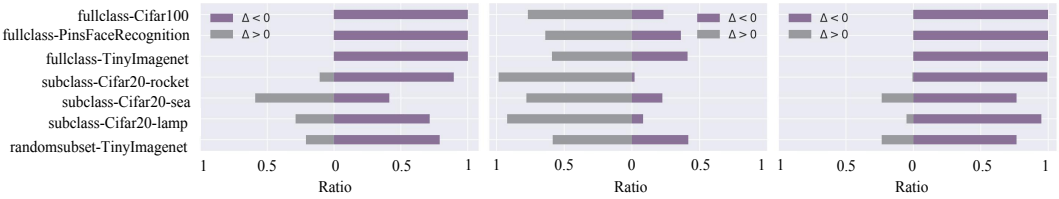

Figure 4: **Ratio of input sensitivity difference Δ rise and fall of the forgetting data under different unlearning settings.** From left to right, Δ is the sample-wise difference between the retrained and pre-trained model on $\|\nabla_{\boldsymbol{x}} f_c\|_F$, $\|\nabla_{\boldsymbol{x}} f_{c'}\|_F$ and $\|\nabla_{\boldsymbol{x}} f_c\|_F - \|\nabla_{\boldsymbol{x}} f_{c'}\|_F$. Sample's contribution to input sensitivity includes promoting $\|\nabla_{\boldsymbol{x}} f_c\|_F$ and suppressing $\|\nabla_{\boldsymbol{x}} f_{c'}\|_F$, thereby enlarging the magnitude gap $\|\nabla_{\boldsymbol{x}} f_c\|_F - \|\nabla_{\boldsymbol{x}} f_{c'}\|_F$.

4 Proposed Method

4.1 MU-Mis: Machine Unlearning by Minimizing Input Sensitivity

In the above section, we theoretically and empirically derived an optimizable and lightweight approximation of sample contributions from the perspective of input sensitivity, showing that they manifest as disproportionately higher sensitivity of the target logit relative to irrelevant logits.

In light of this finding, we propose to withdraw the sample's contribution by reducing such enhancement on the sensitivity magnitude gap. Minimizing this loss guides the pre-trained model to roll back $\|\nabla_x f_c\|_F$ and pick up $\|\nabla_x f_{c'}\|_F$. Mathematically, our proposed unlearning loss is:

$$\mathcal{L}(\mathcal{D}_f; w) = \frac{1}{N_f} \sum_{x_f \in \mathcal{D}_f} (\|\nabla_{\boldsymbol{x}} f_c(x_f, w)\|_F^2 - \|\nabla_{\boldsymbol{x}} f_{c'}(x_f, w)\|_F^2) \tag{3}$$

where N_f is number of the forgetting data, c represents the target class of sample x and $c' \neq c$ denotes an irrelevant class. For each forgetting sample, a new c' is randomly selected every time the loss is computed.

Algorithm 1 MU-Mis: Machine Unlearning by Minimizing Input Sensitivity

Input: Forgetting data $\mathcal{D}_f$; Pre-trained model weights w_p; Learning rate η; Stopping threshold ratio δ.

 # Initialization

$w_0 \leftarrow w_p$, $\epsilon \leftarrow \infty$

 # Iterative optimization

repeat

  for each forgetting sample $x \in \mathcal{D}_f$ **do**

    Randomly select an irrelevant class $c' \neq c$

  end for

  Compute loss $\mathcal{L}$ according to equation 3

  Update $w_{t+1} \leftarrow w_t - \eta \nabla \mathcal{L}$

  Update $\epsilon \leftarrow \min(\epsilon, \|\nabla_x f_{c'}(x, w_t)\|_F)$

until $\|\nabla_x f_{c'}(x, w_t)\|_F > \epsilon$ **and** $\frac{\|\nabla_x f_{c'}(x, w_t)\|_F}{\|\nabla_x f_{c'}(x, w_0)\|_F} > \delta$

Output: Updated model weights w_t

Stopping Guideline. To ensure a practical deployment of MU-Mis, we design a stopping rule for terminating optimization once the withdrawal is completed. Empirical analysis in Appendix F reveals a consistent trend of metrics during our optimization: as the MU-Mis loss decreases, forgetting accuracy (FA) drops steadily, while the accuracies on retained (RA) and test data (TA) initially decline slightly and then grow with the recovery of irrelevant-class logit sensitivity. Crucially, RA approaches the retrained model when this sensitivity returns to its initial level. Therefore, we introduce a threshold ratio δ to govern the termination of unlearning. This criterion ensures that optimization halts when irrelevant-class sensitivity is sufficiently restored. The overall algorithm is outlined in Algorithm 1.

5 Experiments

5.1 Experiment Setups

Tasks, Datasets and Models. We evaluate unlearning across 3 settings: full-class (CIFAR-100 (Krizhevsky et al., 2009), PinsFaceRecognition (Burak, 2020), and Tiny ImageNet (Le & Yang, 2015)), sub-class (CIFAR-20 (Krizhevsky et al., 2009)), and random-subset (CIFAR-10 (Krizhevsky et al., 2009) and SVHN (Netzer et al., 2011)). ResNet-18 (He et al., 2016) is adopted as the default backbone, and we additionally evaluate under ViT (Dosovitskiy et al., 2021) to highlight the efficiency of remaining-data-free methods. Beyond unlearning utility, we assess the resilience of unlearning methods by executing multiple full-class and sub-class unlearning requests iteratively.

Evaluation Metrics. MU methods should be assessed from three aspects (Xu et al., 2024): *utility*, *privacy*, and *efficiency*. For *utility*, we compute forgetting data accuracy (**FA**), remaining data accuracy (**RA**), and test data accuracy (**TA**) of the unlearned model. The average gap (**Avg. Gap**) between the retrained model and the unlearned model across above 3 accuracy-related metrics are computed to illustrate the utility disparity. We compute the train (**FGTA**) and valid (**FGVA**) accuracy on the forgotten classes in sequential unlearning. Regarding the *privacy* guarantee, we use membership inference attack (**MIA**) (Chundawat et al., 2023) to probe the remaining information of the forgetting data. For *efficiency*, we provide the run time efficiency (**RTE**) in **seconds** to indicate timeliness.

Baselines. We compare against 6 remaining-data methods: Bad Teacher(BT) (Chundawat et al., 2023), Fine-tune(FT) (Warnecke et al., 2023), SCRUB (Kurmanji et al., 2024), SSD (Foster et al., 2024b), DUCK (Cotogni et al., 2023) and SalUn (Fan et al., 2024) as well as 4 remaining-data-free methods: RL (Golatkar et al., 2020),NG (Thudi et al., 2022), JiT (Foster et al., 2024a) , SCAR (Bonato et al., 2025). Notably, unlike SCAR, our method requires no auxiliary OOD data. Further details on sequential unlearning settings, metrics, and baselines are provided in Appendix G.1.

5.2 Unlearning Utility

Table 1: Performance overview for **full class** unlearning task evaluated on CIFAR-100 and Tiny ImageNet using ResNet-18. This table includes performances of our proposed MU-Mis, 6 remaining-data-dependent and 4 remaining-data-free methods, which are delineated by a horizontal line. The result format is given by $a_{\pm b}$ with mean a and standard deviation b over 5 independent trials. The metric *average gap (Avg. Gap)* is calculated by the average of the performance gaps measured in accuracy-related metrics, including FA, RA and TA. RTE is reported in **seconds**. Values in terms of accuracy-related metrics deviating by more than 5% from the retrain model are highlighted in red.

Method	CIFAR-100						Tiny ImageNet					
	RA	FA	TA	Avg. Gap↓	MIA	RTE	RA	FA	TA	Avg. Gap↓	MIA	RTE
Pretrain	76.41	79.69	76.47	26.84	95.80	-	65.85	62.00	65.50	21.03	93.59	-
Retrain	$76.52_{\pm 0.27}$	$0.00_{\pm 0.00}$	$75.76_{\pm 0.24}$	0.00	$2.87_{\pm 0.46}$	-	$65.36_{\pm 0.03}$	$0.00_{\pm 0.03}$	$64.90_{\pm 0.03}$	0.00	$4.80_{\pm 0.04}$	-
BT	$76.67_{\pm 0.03}$	$0.00_{\pm 0.00}$	$76.02_{\pm 0.03}$	0.14	$0.00_{\pm 0.00}$	32	$64.90_{\pm 0.01}$	$0.00_{\pm 0.00}$	$64.53_{\pm 0.01}$	0.28	$0.00_{\pm 0.00}$	240
FT	$76.67_{\pm 0.21}$	$0.28_{\pm 0.62}$	$75.88_{\pm 0.22}$	0.19	$0.28_{\pm 0.00}$	250	$64.16_{\pm 0.26}$	$0.00_{\pm 0.00}$	$63.87_{\pm 0.22}$	0.74	$4.40_{\pm 0.58}$	262
SCRUB	$76.81_{\pm 0.04}$	$0.00_{\pm 0.00}$	$76.02_{\pm 0.04}$	0.18	$5.57_{\pm 0.34}$	124	$65.06_{\pm 0.04}$	$0.00_{\pm 0.00}$	$64.69_{\pm 0.03}$	0.17	$14.60_{\pm 0.52}$	860
SSD	$76.27_{\pm 0.00}$	$0.00_{\pm 0.00}$	$75.49_{\pm 0.00}$	0.17	$0.00_{\pm 0.00}$	26	$65.58_{\pm 0.00}$	$0.00_{\pm 0.00}$	$65.19_{\pm 0.00}$	0.17	$0.00_{\pm 0.00}$	59
DUCK	$75.82_{\pm 0.18}$	$0.20_{\pm 0.45}$	$75.13_{\pm 0.17}$	0.51	$0.00_{\pm 0.00}$	100	$64.97_{\pm 0.14}$	$0.00_{\pm 0.00}$	$64.61_{\pm 0.14}$	0.23	$2.60_{\pm 0.46}$	55
SalUn	$76.63_{\pm 0.03}$	$1.20_{\pm 0.45}$	$75.85_{\pm 0.03}$	0.47	$0.00_{\pm 0.00}$	254	$65.21_{\pm 0.10}$	$0.00_{\pm 0.00}$	$64.88_{\pm 0.10}$	**0.06**	$4.40_{\pm 0.40}$	2630
NG	$69.76_{\pm 0.01}$	$0.00_{\pm 0.00}$	$69.23_{\pm 0.01}$	4.43	$0.00_{\pm 0.00}$	2	$59.62_{\pm 0.00}$	$0.00_{\pm 0.00}$	$59.26_{\pm 0.00}$	3.79	$1.80_{\pm 0.00}$	3
RL	$65.98_{\pm 0.12}$	$5.22_{\pm 0.45}$	$65.52_{\pm 0.11}$	8.66	$0.00_{\pm 0.00}$	12	$53.41_{\pm 0.00}$	$0.00_{\pm 0.00}$	$53.04_{\pm 0.01}$	7.94	$2.00_{\pm 0.00}$	10
SCAR	$71.33_{\pm 0.12}$	$5.61_{\pm 0.89}$	$70.66_{\pm 0.14}$	5.29	$13.28_{\pm 0.67}$	367	$59.98_{\pm 0.06}$	$0.00_{\pm 0.00}$	$59.62_{\pm 0.06}$	3.55	$0.67_{\pm 0.12}$	1052
JiT	$65.44_{\pm 0.14}$	$3.00_{\pm 0.76}$	$64.87_{\pm 0.13}$	8.32	$4.44_{\pm 0.30}$	15	$53.82_{\pm 0.09}$	$0.00_{\pm 0.00}$	$53.16_{\pm 0.08}$	7.76	$5.29_{\pm 0.25}$	5
MU-Mis	$76.42_{\pm 0.07}$	$0.00_{\pm 0.00}$	$75.64_{\pm 0.07}$	**0.07**	$0.00_{\pm 0.00}$	30	$64.95_{\pm 0.00}$	$0.00_{\pm 0.00}$	$64.85_{\pm 0.00}$	0.15	$0.20_{\pm 0.00}$	83

Table 2: Performance overview for **sub-class** unlearning task evaluated on 'Rocket' and 'Sea' (where the retrain model exhibits different degrees of generalization ability on the unlearned sub-class) of CIFAR-20 using ResNet-18. The content format follows Table 1.

Method	Rocket						Sea					
	RA	FA	TA	Avg. Gap↓	MIA	RTE	RA	FA	TA	Avg. Gap↓	MIA	RTE
Pretrain	85.26	80.73	85.21	26.53	92.89	-	85.09	97.66	85.21	5.94	91.81	-
Retrain	$84.85_{\pm 0.09}$	$2.69_{\pm 0.45}$	$84.07_{\pm 0.10}$	0.00	$12.06_{\pm 0.75}$	-	$84.60_{\pm 0.22}$	$80.93_{\pm 2.20}$	$84.61_{\pm 0.19}$	0.00	$51.61_{\pm 3.60}$	-
BT	$85.24_{\pm 0.02}$	$2.80_{\pm 0.45}$	$84.36_{\pm 0.02}$	0.26	$0.00_{\pm 0.00}$	27	$82.51_{\pm 0.00}$	$81.00_{\pm 0.00}$	$82.63_{\pm 0.00}$	1.38	$15.00_{\pm 0.00}$	47
FT	$82.70_{\pm 0.19}$	$4.20_{\pm 1.30}$	$81.97_{\pm 0.12}$	1.92	$5.40_{\pm 1.04}$	138	$82.36_{\pm 0.29}$	$88.00_{\pm 1.41}$	$82.43_{\pm 1.60}$	3.83	$58.08_{\pm 1.79}$	417
SCRUB	$84.73_{\pm 0.13}$	$5.80_{\pm 1.30}$	$83.84_{\pm 0.13}$	1.15	$13.28_{\pm 0.02}$	113	$84.86_{\pm 0.10}$	$88.17_{\pm 1.72}$	$84.86_{\pm 0.13}$	2.58	$57.07_{\pm 1.71}$	113
SSD	$84.23_{\pm 0.05}$	$2.60_{\pm 0.89}$	$83.35_{\pm 0.06}$	0.48	$3.76_{\pm 0.36}$	18	$84.79_{\pm 0.00}$	$78.00_{\pm 0.00}$	$84.61_{\pm 0.00}$	1.24	$8.00_{\pm 0.00}$	7
DUCK	$82.09_{\pm 0.33}$	$19.4_{\pm 3.28}$	$81.43_{\pm 0.35}$	7.37	$32.84_{\pm 1.57}$	58	$80.95_{\pm 0.19}$	$66.45_{\pm 2.30}$	$80.77_{\pm 0.19}$	7.34	$54.92_{\pm 2.29}$	68
SalUn	$84.82_{\pm 0.06}$	$2.99_{\pm 1.25}$	$84.00_{\pm 0.05}$	**0.13**	$0.00_{\pm 0.00}$	1042	$82.85_{\pm 0.00}$	$81.00_{\pm 0.00}$	$83.10_{\pm 0.00}$	1.11	$13.40_{\pm 0.00}$	63
NG	$62.84_{\pm 5.66}$	$5.67_{\pm 4.08}$	$62.48_{\pm 5.59}$	15.52	$72.70_{\pm 21.80}$	4	$80.95_{\pm 0.00}$	$75.00_{\pm 0.00}$	$80.84_{\pm 0.02}$	4.45	$60.00_{\pm 0.00}$	3
RL	$60.89_{\pm 1.96}$	$6.52_{\pm 1.07}$	$60.50_{\pm 2.01}$	17.11	$3.70_{\pm 6.51}$	5	$80.48_{\pm 0.02}$	$77.00_{\pm 0.00}$	$80.34_{\pm 0.02}$	4.11	$48.70_{\pm 0.11}$	3
SCAR	$76.49_{\pm 0.22}$	$43.81_{\pm 4.44}$	$76.26_{\pm 0.23}$	19.09	$28.04_{\pm 1.67}$	442	$76.30_{\pm 0.15}$	$77.40_{\pm 2.71}$	$76.12_{\pm 0.19}$	6.77	$51.84_{\pm 1.98}$	434
JiT	$59.15_{\pm 0.05}$	$4.00_{\pm 0.00}$	$58.60_{\pm 0.05}$	17.49	$29.03_{\pm 0.20}$	4	$51.48_{\pm 0.04}$	$7.20_{\pm 1.10}$	$51.04_{\pm 0.04}$	46.81	$32.20_{\pm 0.24}$	4
MU-Mis	$84.28_{\pm 0.18}$	$2.91_{\pm 1.02}$	$83.50_{\pm 0.19}$	0.49	$0.07_{\pm 0.25}$	21	$84.35_{\pm 0.03}$	$81.00_{\pm 2.95}$	$84.33_{\pm 0.05}$	**0.20**	$1.25_{\pm 1.85}$	10

MU-Mis outperforms existing remaining-data-free methods significantly and remains highly competitive with SoTA remaining-data-dependent methods. Table 1 and Table 2 correspond to MU performances on full-class and sub-class unlearning respectively. More experiment results are referred to Appendix H.1. In terms of unlearnig *utility*, MU-Mis achieves the smallest Avg. Gap in full-class-CIFAR-100, full-class-PinsFaceRecognition, sub-class-Sea and sub-class-Lamp unlearning, outperforming all the baseline methods. From the highlighted values in red in the tables, we could

see that existing remaining-data-free methods suffer from poor utility preservation. In terms of *privacy*, MIA values of MU-Mis remain consistently low across all three tasks, indicating successful privacy protection. In addition to resolving the issue of constrained access to the remaining data, our remaining-data-free method also offers a notable advantage in MU efficiency. In unlearning a full class Tiny ImageNet, MU-Mis is up to 30× faster than SalUn, with only 0.09 higher Avg. Gap.

Efficiency advantage is more pronounced on larger scale models. Table 1 shows the performance when unlearning a full class of Tiny ImageNet under ViT (Dosovitskiy et al., 2021). MU-Mis outperforms other remaining-data-free methods [1] significantly and performs comparably with the most competitive method SalUn in terms of model utility and privacy. The efficiency advantage of MU-Mis becomes markedly pronounced: the unlearning time is reduced from more than **1 hour** to **3 minutes**. We also evaluate subclass-CIFAR20-sea unlearning under ViT and show the results in Table A9, where MU-Mis exhibits the best Avg.Gap and is 20× faster than SalUn.

Table 3: Performance overview for **full class** unlearning task evaluated on **Tiny ImageNet** using **ViT**. RTE is reported in **minute**.

Methods	RA	FA	TA	Avg. Gap ↓	MIA	RTE (min)
Pretrain	84.21	87.5	84.23	30.44	95.40	-
Retrain	$86.35_{\pm 0.17}$	$0.00_{\pm 0.00}$	$85.92_{\pm 0.14}$	0.00	$8.80_{\pm 0.26}$	-
Salun	$83.94_{\pm 0.16}$	$0.00_{\pm 0.00}$	$83.49_{\pm 0.14}$	1.88	$0.00_{\pm 0.00}$	81
NG	$63.12_{\pm 0.00}$	$0.00_{\pm 0.00}$	$62.88_{\pm 0.00}$	15.28	$0.00_{\pm 0.00}$	0.21
RL	$67.69_{\pm 0.00}$	$0.00_{\pm 0.00}$	$67.43_{\pm 0.00}$	12.38	$0.00_{\pm 0.00}$	0.15
MU-Mis	$82.13_{\pm 0.24}$	$0.00_{\pm 0.00}$	$82.17_{\pm 0.23}$	2.69	$0.00_{\pm 0.00}$	3

5.3 Unlearning Resilience: Sequential Machine Unlearning

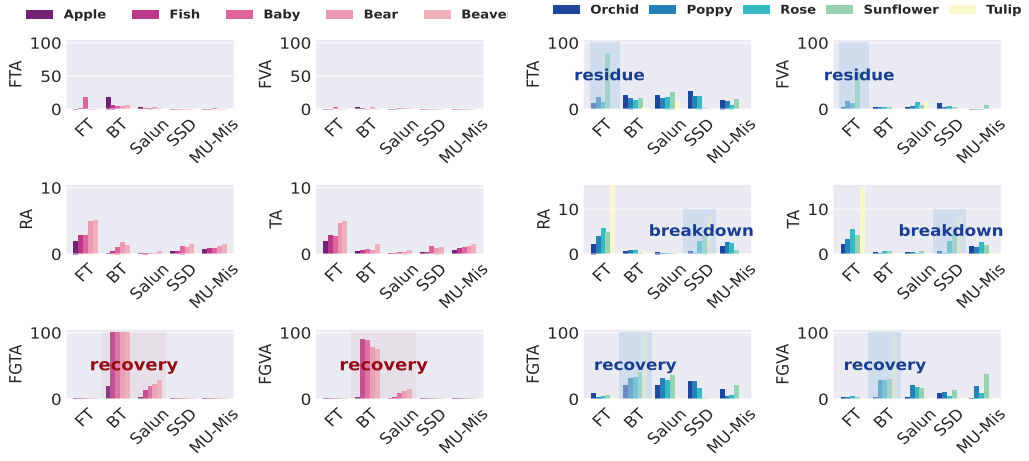

Figure 5: **Disparities in accuracy-related metrics between the unlearned model and the retrained model for full class and sub-class sequential unlearning.** Left: Iteratively unlearns 5 distinct full classes of CIFAR-100. Right: Iteratively unlearns 5 sub-classes of the same super-class 'Flower'.

Sequential unlearning requires principled unlearning mechanisms. In practice, unlearning requests may arrive sequentially, requiring multiple executions of the unlearning method. Wang et al. (2024a) point out that sequential unlearning greatly challenges the memorization management ability of unlearning methods due to underlying associations among unlearned classes. The sequentially unlearned model might break down due to disordered forgetting operation, exposing its accumulated effects on model knowledge. Therefore, to highlight the importance of principled forgetting, we perform sequential unlearning. We examine the impact of subsequent requests on previous unlearning efforts and present the disparities between the unlearned model and the retrain model at each iteration in accuracy-related MU metrics in Figure 5. For detailed experiment settings, refer to Appendix G.1.

Dificiencies in existing MU methods. From Figure 5, we can see that there are 3 kinds of deficiencies in existing SoTA MU methods:

(i) Performance Recovery. The performance on the forgotten classes stages a recovery in BT and Salun unlearned model, indicated by the above zero FGTA and FGVA. This suggests that retargeting model's outputs of the forgetting data does not completely remove associated knowledge, posing a substantial risk since the concealed information might still be exploited by privacy attackers.

[1] We failed to identify effective hyper-parameters for JiT and SCAR for this experiment.

(ii) Knowledge Residue. High disparity of FTA and FVA in sub-class task indicates that FT method, which relies on "catastrophic forgetting" (Kirkpatrick et al., 2017) to unlearn, fails to unlearn effectively in sub-class task due to the resemblance between the forgetting and remaining data.

(iii) Utility Breakdown. In sub-class task, SSD exhibits a marked decline in utility after the last unlearning request, demonstrated by the final RA of 76.33%. In contrast, RA in the retrained model and MU-Mis unlearned model are respectively 84.83% and 84.59%. Such a plummet implies a potential risk of model utility breakdowns when the magnitude of parameters is continuously scaled.

Resilient performance of MU-Mis to sequential unlearning requests. To facilitate an intuitive assessment in terms of utility and resilience, we compute the utility Avg. Gap and resilience Avg. Gap for each iteration in Figure 6. The utility Avg. Gap is averaged over FTA, FVA, RA and TA, and the resilience Avg. Gap is averaged over FGTA and FGVA. From Figure 6, it is evident that MU-Mis and SSD are significantly better than BT, FT, and Salun, demonstrating a notably small disparity to the retrained model regarding both the utility and resilience Avg. Gap across the full class and sub-class tasks. Importantly, MU-Mis achieves these results without relying on the remaining data, which are required by SSD.

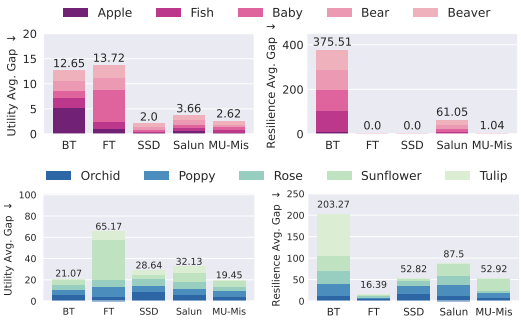

Figure 6: **Overview of utility Avg. Gap and resilience Avg. Gap during full class (upper) and sub-class (bottom) sequential unlearning.**

Minimal KL divergence of MU-Mis from retrain model during sequential unlearn. Beyond model predictions, we further provide the empirical KL divergence (introduced in Appendix G.4) between the unlearned model and the retrained model's output distributions during sequential requests in Figure 7. It is evident that MU-Mis exhibits the lowest KL divergence from the retrained model throughout both the full-class and sub-class sequential unlearning processes.

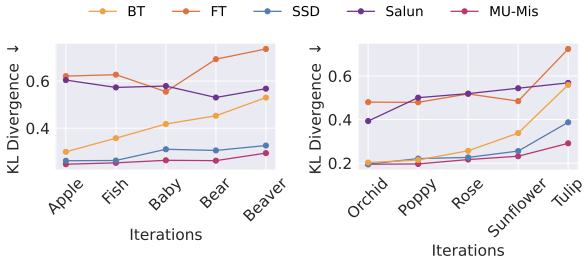

Figure 7: **KL divergence between outputs of unlearned model and retrained model during full class (left) and sub-class (right) sequential unlearning.**

Summary. In general, MU-Mis stands out with its comprehensive capabilities in terms of unlearning utility, unlearning resilience as well as output indistinguishability, while current SoTA MU methods are disclosed to exhibit limitations and deficiencies in certain aspects. Their inappropriate or inadequate unlearning approaches undermine their reliability and applicability in practical scenarios.

5.4 Supplementary Experiments and Analyses

We provide the following experiments and analyses for completeness in Appendix: *(i)* an ablation study of each loss term in MU-Mis in Appendix H.2; *(ii)* a hyper-parameter sensitivity analysis showing stability of MU-Mis in Appendix H.3; *(iii)* visualizations of attention map confirming effectiveness of MU-Mis in Appendix H.4; *(iv)* an empirical analysis attributing the effectiveness of MU-Mis to the orthogonality of input sensitivity gradients among samples in Appendix I.

6 Conclusion

There are 3 main challenges in machine unlearning: *the stochasticity of training*, *incrementality of training*, and *catastrophe of unlearning* (Wang et al., 2024b). We address incrementality by quantifying sample contribution through the lens of input sensitivity. Building on this, our proposed MU-Mis achieves effective and efficient unlearning without compromising model utility, alleviating catastrophic unlearning. Experiments validate the superiority of this principled forgetting mechanism. Overall, MU-Mis is well-grounded, lightweight and remaining-data-free, offering a practical and competitive alternative to existing unlearning methods. Furthermore, we highlight in Appendix J that there is a profound connection between input sensitivity view and machine unlearning.

BIBLIOGRAPHY

Juhan Bae, Nathan Ng, Alston Lo, Marzyeh Ghassemi, and Roger B Grosse. If influence functions are the answer, then what is the question? *Advances in Neural Information Processing Systems (NeurIPS)*, 35:17953–17967, 2022. 14

Samyadeep Basu, Phil Pope, and Soheil Feizi. Influence functions in deep learning are fragile. In *International Conference on Learning Representations (ICLR)*, 2020. 14

Jacopo Bonato, Marco Cotogni, and Luigi Sabetta. Is retain set all you need in machine unlearning? restoring performance of unlearned models with out-of-distribution images. In *European Conference on Computer Vision (ECCV)*, pp. 1–19. Springer, 2025. 7, 15, 17

Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pp. 141–159. IEEE, 2021. 14

Burak. Pinterest face recognition dataset. www.kaggle.com/datasets/hereisburak/pins-facerecognition, 2020. 6, 16

Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *IEEE Symposium on Security and Privacy*, pp. 463–480. IEEE, 2015. 1

Nicholas Carlini, Matthew Jagielski, Chiyuan Zhang, Nicolas Papernot, Andreas Terzis, and Florian Tramer. The privacy onion effect: Memorization is relative. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:13263–13276, 2022. 16

Yuanyuan Chen, Boyang Li, Han Yu, Pengcheng Wu, and Chunyan Miao. Hydra: Hypergradient data relevance analysis for interpreting deep neural networks. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 35, pp. 7081–7089, 2021. 14

Andreas Christmann and Ingo Steinwart. *Support Vector Machines*. Springer, 2008. 2

Vikram S Chundawat, Ayush K Tarun, Murari Mandal, and Mohan Kankanhalli. Can bad teaching induce forgetting? unlearning in deep networks using an incompetent teacher. In *AAAI Conference on Artificial Intelligence (AAAI)*, pp. 7210–7217, 2023. 1, 7, 15, 16, 17, 22

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning (ML)*, 20:273–297, 1995. 2

Marco Cotogni, Jacopo Bonato, Luigi Sabetta, Francesco Pelosin, and Alessandro Nicolosi. Duck: Distance-based unlearning via centroid kinematics. *arXiv preprint arXiv:2312.02052*, 2023. 7, 17

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. In *9th International Conference on Learning Representations (ICLR)*, 2021. 6, 8

Harris Drucker and Yann Le Cun. Improving generalization performance using double backpropagation. *IEEE Transactions on Neural Networks*, 3(6):991–997, 1992. 22

Chongyu Fan, Jiancheng Liu, Yihua Zhang, Eric Wong, Dennis Wei, and Sijia Liu. Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation. In *International Conference on Learning Representations (ICLR)*, 2024. 1, 7, 15, 17, 22

Vitaly Feldman. Does learning require memorization? a short tale about a long tail. In *the Annual ACM SIGACT Symposium on Theory of Computing*, pp. 954–959, 2020. 1, 4

Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. *Advances in Neural Information Processing Systems (NeurIPS)*, 33: 2881–2891, 2020. 1, 4

Jack Foster, Kyle Fogarty, Stefan Schoepf, Cengiz Öztireli, and Alexandra Brintrup. Zero-shot machine unlearning at scale via lipschitz regularization. *arXiv preprint arXiv:2402.01401*, 2024a. 7, 15, 17, 22

Jack Foster, Stefan Schoepf, and Alexandra Brintrup. Fast machine unlearning without retraining through selective synaptic dampening. In *AAAI Conference on Artificial Intelligence (AAAI)*, pp. 12043–12051, 2024b. 7, 15, 17

Isha Garg, Deepak Ravikumar, and Kaushik Roy. Memorization through the lens of curvature of loss function around samples. In *International Conference on Machine Learning, (ICML)*, 2024. 23

Ryan Giordano, William Stephenson, Runjing Liu, Michael Jordan, and Tamara Broderick. A swiss army infinitesimal jackknife. In *International Conference on Artificial Intelligence and Statistics*, pp. 1139–1147. PMLR, 2019. 14

Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9304–9312, 2020. 1, 7, 17

Will Grathwohl, Kuan-Chieh Wang, Joern-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. In *International Conference on Learning Representations (ICLR)*. 5

Laura Graves, Vineel Nagisetty, and Vijay Ganesh. Amnesiac machine learning. In *AAAI Conference on Artificial Intelligence (AAAI)*, pp. 11516–11524, 2021. 1, 14, 15, 22

Chuan Guo, Tom Goldstein, Awni Y. Hannun, and Laurens van der Maaten. Certified data removal from machine learning models. In *International Conference on Machine Learning (ICML)*, pp. 3832–3842, 2020. 14

Zayd Hammoudeh and Daniel Lowd. Training data influence analysis and estimation: A survey. *Machine Learning (ML)*, 113(5):2351–2403, 2024. 14

Satoshi Hara, Atsushi Nitanda, and Takanori Maehara. Data cleansing for models trained with sgd. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019. 14

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016. 6, 16

Zhengbao He, Tao Li, Xinwen Cheng, Zhehao Huang, and Xiaolin Huang. Towards natural machine unlearning. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2025. 2

Tuan Hoang, Santu Rana, Sunil Gupta, and Svetha Venkatesh. Learn to unlearn for deep neural networks: Minimizing unlearning interference with gradient projection. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 4819–4828, 2024. 15

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114 (13):3521–3526, 2017. 9

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning (ICML)*, pp. 1885–1894. PMLR, 2017. 14

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6, 16

Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. Towards unbounded machine unlearning. *Advances in Neural Information Processing Systems (NeurIPS)*, 36, 2024. 1, 7, 15, 17, 22

Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 6, 16

Shen Lin, Xiaoyu Zhang, Chenyang Chen, Xiaofeng Chen, and Willy Susilo. Erm-ktp: Knowledge-level machine unlearning via knowledge transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20147–20155, 2023. 15, 22

Jiancheng Liu, Parikshit Ram, Yuguang Yao, Gaowen Liu, Yang Liu, PRANAY SHARMA, Sijia Liu, et al. Model sparsity can simplify machine unlearning. *Advances in Neural Information Processing Systems (NeurIPS)*, 36, 2024. 15, 16, 22

Ronak Mehta, Sourav Pal, Vikas Singh, and Sathya N Ravi. Deep unlearning via randomized conditionally independent hessians. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10422–10431, 2022. 14

Xianjia Meng, Yong Yang, Ximeng Liu, and Nan Jiang. Active forgetting via influence estimation for neural networks. *International Journal of Intelligent Systems*, 37(11):9080–9107, 2022. 14

Fan Mo, Anastasia Borovykh, Mohammad Malekzadeh, Soteris Demetriou, Deniz Gündüz, and Hamed Haddadi. Quantifying and localizing usable information leakage from neural network gradients. *arXiv preprint arXiv:2105.13929*, 2021. 23

Seth Neel, Aaron Roth, and Saeed Sharifi-Malvajerdi. Descent-to-delete: Gradient-based methods for machine unlearning. In *Algorithmic Learning Theory*, pp. 931–962. PMLR, 2021. 14, 22

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, pp. 7. Granada, Spain, 2011. 6, 16

Vardan Papyan. Traces of class/cross-class structure pervade deep learning spectra. *Journal of Machine Learning (JMLR)*, 21(1):10197–10260, 2020. 22

Vardan Papyan, XY Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences (PNAS)*, 117 (40):24652–24663, 2020. 22

Alexandra Peste, Dan Alistarh, and Christoph H Lampert. SSSE: Efficiently erasing samples from trained machine learning models. In *NeurIPS 2021 Workshop Privacy in Machine Learning*, 2021. 14

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:19920–19930, 2020. 14

Deepak Ravikumar, Efstathia Soufleri, Abolfazl Hashemi, and Kaushik Roy. Unveiling privacy, memorization, and input curvature links. In *International Conference on Machine Learning, (ICML)*, 2024. 23

General Data Protection Regulation. General data protection regulation (GDPR). *Intersoft Consulting, Accessed in October*, 24(1), 2018. 1

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *IEEE International Conference on Computer Vision (ICCV)*, pp. 618–626, 2017. 22

Thanveer Shaik, Xiaohui Tao, Haoran Xie, Lin Li, Xiaofeng Zhu, and Qing Li. Exploring the landscape of machine unlearning: A survey and taxonomy. *arXiv preprint arXiv:2305.06360*, 1(2), 2023. 14

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18. IEEE, 2017. 1

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017. 22

Suraj Srinivas and François Fleuret. Rethinking the role of gradient-based attribution methods for model interpretability. In *9th International Conference on Learning Representations (ICLR)*, 2021. 5

Vinith Suriyakumar and Ashia C Wilson. Algorithms that approximate data removal: New results and limitations. *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 18892–18903, 2022. 14

Ayush K Tarun, Vikram S Chundawat, Murari Mandal, and Mohan Kankanhalli. Fast yet effective machine unlearning. *IEEE Transactions on Neural Networks and Learning Systems*, 2023. 15

Anvith Thudi, Gabriel Deza, Varun Chandrasekaran, and Nicolas Papernot. Unrolling sgd: Understanding factors influencing machine unlearning. In *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*, pp. 303–319. IEEE, 2022. 1, 7, 17, 22

Hanling Tian, Yuhang Liu, Mingzhen He, Zhengbao He, Zhehao Huang, Ruikai Yang, and Xiaolin Huang. Simulating training dynamics to reconstruct training data from deep neural networks. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025. 1

Cheng-Long Wang, Qi Li, Zihang Xiang, and Di Wang. Has approximate machine unlearning been evaluated properly? from auditing to side effects. *arXiv preprint arXiv:2403.12830*, 2024a. 8, 16

Weiqi Wang, Zhiyi Tian, and Shui Yu. Machine unlearning: A comprehensive survey. *arXiv preprint arXiv:2405.07406*, 2024b. 1, 2, 9

Alexander Warnecke, Lukas Pirch, Christian Wressnegger, and Konrad Rieck. Machine unlearning of features and labels. In *30th Annual Network and Distributed System Security Symposium, NDSS*, 2023. 7, 17

Ga Wu, Masoud Hashemi, and Christopher Srinivasa. Puma: Performance unchanged model augmentation for training data removal. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 36, pp. 8675–8682, 2022. 14

Yinjun Wu, Edgar Dobriban, and Susan Davidson. Deltagrad: Rapid retraining of machine learning models. In *International Conference on Machine Learning (ICML)*, pp. 10355–10366. PMLR, 2020. 14, 22

Jie Xu, Zihan Wu, Cong Wang, and Xiaohua Jia. Machine unlearning: Solutions and challenges. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 8(3):2150–2168, 2024. 7, 14, 16

Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. idlg: Improved deep leakage from gradients. *arXiv preprint arXiv:2001.02610*, 2020. 23

Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019. 1, 23

APPENDIX

A ETHICS STATEMENT

This work studies machine unlearning (MU), motivated by the "right to be forgotten," with the goal of enhancing user privacy and data protection. All experiments are conducted on publicly available datasets and standard benchmark models; no sensitive or personally identifiable information is used. While unlearning techniques could in principle be misused to manipulate model behavior, our focus is on strengthening trust and accountability in machine learning systems. We believe this work contributes positively to the development of privacy-preserving and ethically responsible AI.

B REPRODUCIBILITY STATEMENT

We include anonymized supplementary materials containing the complete algorithm implementations for executing all experiments. We provide detailed experimental settings, hyperparameters, datasets, and evaluation metrics in our Appendix to ensure reproducibility.

C THE USE OF LLMS

Large language models (LLMs) were employed solely as auxiliary writing tools. Their usage was strictly limited to surface-level assistance, including grammar correction, stylistic polishing, clarity improvement, and formatting consistency. LLMs were not involved in formulating research ideas, designing methods, conducting analyses, interpreting results, or drawing conclusions. At no stage were LLMs used to generate original content, experimental designs, or theoretical claims. All text segments refined with LLM assistance were subsequently reviewed, validated, and, where necessary, rewritten by the authors to ensure technical accuracy and precision of expression. The authors bear full responsibility for the final presentation and content of this paper. This disclosure is made in accordance with conference guidelines on LLM usage to ensure transparency and research integrity.

D RELATED WORK

The primary goal of machine unlearning (MU) (Shaik et al., 2023; Xu et al., 2024; Bourtoule et al., 2021) is to remove the influence of specific data points on a pre-trained model, protecting data privacy. MU can be categorized into two types (Shaik et al., 2023): *exact* MU and *approximate* MU. Exact MU approaches *parameters* of the retrained model and guarantees the privacy risk statistically, extensively studied within convex settings and linear models (Guo et al., 2020; Suriyakumar & Wilson, 2022; Neel et al., 2021; Giordano et al., 2019; Koh & Liang, 2017). However, exact MU for over-parameterized deep models requires retraining the model from scratch, which is impractical. Therefore, to balance the unlearning completeness and efficiency, approximate MU is proposed to approach the *output distribution* of the retrained model. In this paper, we concentrate on approximate unlearning, as it is more practical in large-scale models and situations with limited time and resources.

MU by Gradient-Based Update. One straightforward way to retrieve sample contribution is to keep and utilize the historical information (e.g. parameters and gradients) during the training process. Graves et al. (Graves et al., 2021) withdraw gradient updates of related batches, and Wu et al. (Wu et al., 2020) utilize intermediate checkpoints and quasi-newton method for rapid retraining. The requirement of storing historical information raises memory concerns. Another line of work estimates the contribution of the forgetting data on learned model parameters through influence function (Koh & Liang, 2017), initially introduced to unlearning by Guo et al. (Guo et al., 2020). However, calculating the inverse Hessian in influence function is computationally expensive for DNNs and follow-up studies are devoted to reducing the computation (Mehta et al., 2022; Peste et al., 2021; Meng et al., 2022). While influence-based unlearning shows potential, the withdrawal still hurts performance on the remaining data (Wu et al., 2022). Moreover, the influence function is revealed to be fragile in DNNs (Basu et al., 2020; Bae et al., 2022; Hammoudeh & Lowd, 2024) due to its reliance on the assumptions of convexity and optimality. Existing data influence estimations for DNNs (Hara et al., 2019; Pruthi et al., 2020; Chen et al., 2021; Hammoudeh & Lowd, 2024) all require retracing the training trajectory and cannot be optimized and applied to MU. In this paper, we theoretically

indicate that sample contributions will be approximately reflected in the sensitivity of the pre-trained model to input samples, opening up a new perspective to view sample contribution in DNNs.

MU by loss guided re-optimization. Above gradient-based unlearning methods suffer from practical limitations for DNNs. Generally, practical MU methods unlearn by fine-tuning the model to optimize a proposed loss. They typically follow two design ideas: one is to make model's behavior on the forgetting data similar to that on unseen data through knowledge distillation (Chundawat et al., 2023; Lin et al., 2023; Kurmanji et al., 2024) or label confusion (Graves et al., 2021; Fan et al., 2024), the other is to suppress the part of parameters that are responsible for predictions of the forgetting data (Liu et al., 2024; Foster et al., 2024a; Fan et al., 2024). However, both the confusion and suppression approaches inadvertently damage the model's utility on the remaining data. As a result, unlearning is done either in an "impair-then-repair" regime (Tarun et al., 2023) or through specifically designed mechanisms (Hoang et al., 2024; Foster et al., 2024b; Fan et al., 2024) to alleviate the damage. Since additional maintenance with the remaining data undermines the efficiency of MU, we pursue a more principled forgetting operation by explicitly identifying sample contributions. This enables unlearning with minimal utility degradation and eliminates the need for compensatory procedures.

Remaining-data-free MU. Developing remaining-data-free methods aligns more closely with the essence and practical demands of MU, given the limited accessibility of retained data and the need for efficiency in practice. JiT (Foster et al., 2024a) proposes to smooth the output around the forgetting data by minimizing local Lipschitz value. While SCAR (Bonato et al., 2025) distills knowledge from the pre-trained model and utilizes Out-of-distribution (OOD) data as a surrogate to preserve model utility. However, both approaches have an obvious performance gap to SoTA remaining-data-dependent methods, and SCAR still relies on additional OOD data. Remaining-data-free unlearning is essentially about developing a more principled forgetting mechanism. By identifying sample contribution, we enable a nuanced removal of forgetting data and achieve significant improvements over existing methods.

E A Toy Example complementing sample contribution derivation

We use a simple MLP model to illustrate the insight of $\mathcal{S}_k(\hat{x},\tilde{x}) << \mathcal{S}_k(\hat{x},\hat{x})$ with $\hat{x} \neq \tilde{x}$. Assume the l^{th} layer output of model is $x^l = \phi(\theta^l x^{l-1})$, where θ^l refers to l^{th} layer parameter and ϕ refers to activation function. Then,

$$g_k = \frac{\partial f_k}{\partial \theta^l} = \frac{\partial f_k}{\partial(\theta^l x^{l-1})} {x^{l-1}}^T, \tag{A1}$$

$$\begin{aligned} g'_k &= \frac{\partial f_k}{\partial \theta^l \partial x} = \frac{\partial f_k}{\partial(\theta^l x^{l-1})} \frac{\partial {x^{l-1}}^T}{\partial x} \\ &= \frac{\partial f_k}{\partial(\theta^l x^{l-1})} \phi'(\theta^{l-1}x^{l-2}){\theta^{l-1}}^T \frac{\partial x^{l-2}}{\partial x}. \end{aligned} \tag{A2}$$

Thereby, the inner-dot $g'_k(\hat{x})g_k(\tilde{x}) \propto \tilde{x}^{l-1}\phi'(\theta^{l-1}\hat{x}^{l-2})$. If ReLU activation is used, where $\phi'(x) = 1$ if $x > 0$ else $\phi'(x) = 0$, $\mathcal{S}_k(\hat{x},\tilde{x}) \propto g'_k(\hat{x})g_k(\tilde{x})$ will be quite small. The conclusion here is that the residual term is relatively smaller than the contribution term. Therefore, the contribution of a training sample to the training process would be approximately reflected in the pre-trained model's output sensitivity to the sample.

F Stopping Guidance

The optimization should cease once the withdrawal is completed, requiring a stopping guideline for practical use. We monitor both the optimization objective and unlearning metrics during minimizing the MU-Mis loss Eq.(3) in Figure A1, using the example of unlearning with ResNet-18 on fullclass-CIFAR100-rocket. In Figure A1, different colors represent different learning rates and the purple dashed line represents the accuracy of the retrained model. A consistent trend on accuracy change during unlearning is observed across different learning rates: as the optimization of MU-Mis loss progresses, the accuracy of the forgetting data (FA) gradually decreases, the accuracy of the remaining (RA) and test data (TA) first decrease slightly and then grow up with the recovery of $\|\nabla_x f_{c'}(x, w_p)\|_F$.

Notably, when $\|\nabla_x f_{c'}(x, w)\|_F$ recovers close to the level of its initial value $\|\nabla_x f_{c'}(x, w_p)\|_F$, RA approaches the retrained model across different learning rates. There exists a clear relationship between the unlearning progress and model performance, allowing for effective unlearning by stopping the optimization timely. To this end, we introduce a stopping threshold ratio δ to regulate the time of stopping. We record the minimal value of irrelevant class logit sensitivity as ϵ and terminate unlearning process when $\|\nabla_x f_{c'}(\mathcal{D}_f, w)\|_F > \epsilon$ and $\frac{\|\nabla_x f_{c'}(\mathcal{D}_f, w)\|_F}{\|\nabla_x f_{c'}(\mathcal{D}_f, w_p)\|_F} > \delta$.

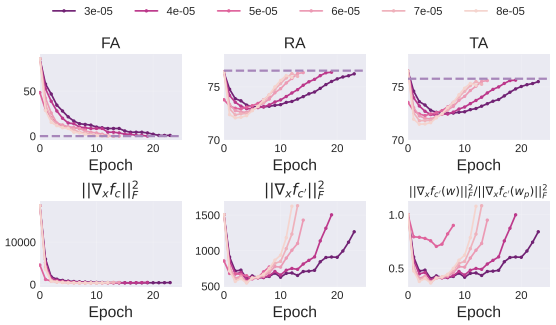

Figure A1: Accuracy and optimization objective during fullclass-CIFAR100-rocket unlearning with different learning rates on ResNet-18. FA decreases gradually, RA and TA first drop slightly and then rise with the recovery of $\|\nabla_x f_{c'}(x, w)\|_F$. The endpoint of each curve corresponds to the time when $\|\nabla_x f_{c'}\|_F$ exceeds 90% of its initial value.

G Experiment Details

G.1 Experiment Setting

Tasks, Datasets and Models. We investigate 3 kinds of unlearning tasks in supervised image classification scenarios, including forgetting a full class, a sub-class under a super-class, and a random subset. We evaluate full class unlearning on CIFAR-100 (Krizhevsky et al., 2009), PinsFaceRecognition (Burak, 2020), and Tiny ImageNet (Le & Yang, 2015), sub-class unlearning on three sub-classes of CIFAR-20 (Krizhevsky et al., 2009), random subset unlearning on CIFAR-10 (Krizhevsky et al., 2009) and SVHN (Netzer et al., 2011). We perform sequential unlearning by iteratively unlearning a full and a sub-class to evaluate the algorithm's robustness to privacy onion effect (Carlini et al., 2022). For iterative full class MU, we iteratively unlearn 5 distinct classes (label 0-4, corresponding to "Apple", "Fish", "Baby", "Bear" and "Beaver") of CIFAR-100 in line with Wang et al. (Wang et al., 2024a). For sub-class setting, we iteratively unlearn 5 sub-classes ("orchid", "poppy", "rose", "sunflower", "tulip") under the same superclass "flower" of CIFAR-20. We use ResNet-18 (He et al., 2016) for all the above experiments. To further indicate the significant efficiency advantage of our remaining-data-free method, we perform full class unlearning on Tiny-ImageNet and sub-class unlearning on CIFAR-20 with ViT.

Evaluation Metrics. MU methods should be assessed from three aspects: *utility*, *privacy*, and *efficiency*(Xu et al., 2024). Beyond that, in practice, where the unlearning requests are made constantly, the unlearning *resilience* should be assessed, i.e. subsequent unlearning should not spoil previous unlearning efforts. For *utility*, we compute forgetting data accuracy (**FA**), remaining data accuracy (**RA**), and test data accuracy (**TA**) of the unlearned model. FA and RA are computed on the valid set in class-wise unlearning and on the train set in random subset unlearning. We compute the average gap (**Avg. Gap**) between the retrained model and the unlearned model on accuracy-related metrics, including FA, RA and TA to illustrate the utility disparity. In terms of *resilience*, we evaluate on sequential unlearning tasks. We compute the train (**FGTA**) and valid (**FGVA**) accuracy on the forgotten classes and quantify the unlearning resilience with the average of their disparity to the retrained model (**Resilience Avg. Gap**). To further examine the indistinguishability between the retrained and unlearned model, we compute the **KL divergence** between their output distributions over the entire dataset. The retrained model is an oracle of approximate MU, therefore, above disparity metrics should be as small as possible. Regarding the *privacy* guarantee, we use membership inference attack (**MIA**) (Chundawat et al., 2023) to probe the remaining information of the forgetting data. The MIA success rate indicates how many samples in $\mathcal{D}_f$ are predicted as membership samples of the unlearned model. From a privacy perspective, a lower MIA value implies less information leakage in the unlearned model and is preferred (Liu et al., 2024). For *efficiency*, we provide the run time efficiency (**RTE**) in **seconds** to indicate timeliness.

Baselines. We compare our method along with 6 baselines which utilize the remaining data, as well as 4 remaining-data-free methods. The 6 baselines include Bad Teacher (**BT**) (Chundawat et al., 2023),Finetune (**FT**) (Warnecke et al., 2023), SCalable Remembering and Unlearning unBound (**SCRUB**) (Kurmanji et al., 2024), Selective-Synaptic-Dampening (**SSD**) (Foster et al., 2024b), Distance-based Unlearning via Centroid Kinematics (**DUCK**) (Cotogni et al., 2023) and Saliency-based unlearning (**SalUn**) (Fan et al., 2024). The 4 remaining-data-free methods include Random Labeling (**RL**) (Golatkar et al., 2020), Negative Gradient (**NG**) (Thudi et al., 2022), Just in Time unlearning (**JiT**) (Foster et al., 2024a) and Selective-distillation for Class and Architecture-agnostic unleaRning (**SCAR**) (Bonato et al., 2025).

The detailed method of each baseline is as the following:

- **BT** (Chundawat et al., 2023): Bad Teacher transfers knowledge from useful and useless teachers for the remaining data and the forgetting data. The code source is https://github.com/if-loops/selective-synaptic-dampening.
- **FT** (Warnecke et al., 2023): Finetune optimizes the pre-trained model with the remaining data, unlearning relying on "catastrohpic forgetting". The code source is https://github.com/if-loops/selective-synaptic-dampening.
- **SCRUB** (Kurmanji et al., 2024): SCRUB aims to push outputs of the student model (the unlearned model) away from the teacher model (the pre-trained model) to distill knowledge. This is achieved by first performing several max-steps (distill the knowledge) and then perform several min-steps (regain performance on the remaining data with cross-entropy loss). The code source is https://github.com/meghdadk/SCRUB.
- **SSD** (Foster et al., 2024b): SSD uses the Fisher information matrix to assess parameter importance and suppress parameters that are important to the forgetting data while less important to the remaining data. The code source is https://github.com/if-loops/selective-synaptic-dampening.
- **DUCK** (Cotogni et al., 2023): DUCK employs metric learning to guide the removal of samples matching the nearest incorrect centroid in the embedding space. The code source is https://github.com/OcraM17/DUCK.
- **SalUn** (Fan et al., 2024): SalUn computes weight saliency map to enable the most important weights for the forgetting data. The code source is https://github.com/OPTML-Group/Unlearn-Saliency.
- **NG** (Thudi et al., 2022): Negative gradient computes several steps of gradient ascent with the forgetting data. The source code is https://github.com/jbonato1/scar.
- **RL** (Golatkar et al., 2020): Random label relabels the forgetting data with randomly assigned class and fine-tune the model with computed cross-entropy loss. The source code is https://github.com/jbonato1/scar.
- **SCAR** (Bonato et al., 2025): SCAR utilizes Out-of-distribution (OOD) data as a surrogate for the remaining data and distills the knowledge of the original model into the unlearned model to preserve model utility. The source code is https://github.com/jbonato1/scar.
- **JiT** (Foster et al., 2024a): JiT smooths the model output around the forgetting data by minimizing the local Lipschitz constant. The source code is https://github.com/jwf40/Information-Theoretic-Unlearning.

G.2 Training Details

For **ResNet-18**, training uses SGD with a momentum of 0.9, weight decay of 5×10^{-4}, and batch size of 128 with a learning rate initialized at 0.1. The learning rate decays at 60,120,160 by 0.1 with a total of 200 epochs.

For **ViT**, we initialize with model pre-trained on ImageNet provided by torchvision. Then we randomly initialize the last fully connected layers and train it with SGD with a momentum of 0.9, weight decay of 5×10^{-4}, batch size 64, constant learning rate $\eta = 0.1$ for 10 epochs. All the experiments are conducted on a single RTX 4090.

G.3 HYPER-PARAMETERS

For **MU-Mis**, we optimize the pretrained model under **model.eval()** mode with **vanilla SGD** without momentum for ResNet-18 and Adam for ViT. For only the forgetting data are used in MU-Mis, we must freeze the batch norm layers to avoid spoiling the remaining data. We use batch size of 256 for MU-Mis across all the experiments with ResNet-18 and batch size of 32 with ViT. We report the learning rate η and stopping threshold δ used in different settings in Table A1 for reproducibility.

For each baseline, We perform grid search to find the best hyper-parameters in each setting. The hyper-parameter sweep range for each method is presented in Table A4. The hyper-parameters used for all the methods in sequential full class and sub-class tasks are shown in Table A2 and Table A3. For all the methods, we fix batch size as 256 for ResNet-18 and 64 for ViT unless otherwise stated in hyper-parameter ranges. For BT, we use constant learning rate. For fine-tune based methods, e.g. FT, as well as SCRUB and SalUn, we use cosine scheduler. We fix temperature$= 1$, alpha $= 0.5$, gamma $= 0.99$, weight decay $= 5 \times 10^{-4}$ for SCRUB. We fix weight decay $= 5 \times 10^{-4}$ for DUCK and SCAR.

Table A1: Hyperparameters of MU-Mis h(learning rate η and stopping threshold ratio δ).

Setting	η	δ
fullclass-CIFAR-100	7×10^{-5}	1.45
fullclass-PinsFaceRecognition	4×10^{-4}	0.68
fullclass-Tiny-ImageNet	4×10^{-6}	1.1
subclass-CIFAR-20-rocket	3×10^{-5}	3.00
subclass-CIFAR-20-sea	2×10^{-5}	0.93
subclass-CIFAR-20-lamp	5×10^{-5}	1.80
fullclass-Tiny-ImageNet-ViT	5×10^{-4}	1.03

Table A2: Hyperparameters for full class sequential unlearning in Fig. 5.

Methods	**Hyperparameters**
Retrain	epoch $= 200$, lr $= 0.1$, milestones $= [60, 120, 160]$.
BT	epoch $= 10$, lr $= \{5, 5, 5, 1, 5\} \times 10^{-5}$, temperature scalar $= \{3, 1, 1, 1, 5\}$.
FT	epoch $= 10$, lr $= 10^{-1}$ for all iterations.
SSD	dampening constant $\lambda = \{1, 1, 1, 1, 0.1\}$, selection weight $\alpha = \{95, 70, 50, 70, 80\}$.
SalUn	epoch $= 10$, lr $= 10^{-3}$, threshold $= 0.6$ for all iterations.
MU-Mis	epoch $= 50$, lr $= \{2, 1, 1, 0.5, 0.8\} \times 10^{-4}$

Table A3: Hyperparameters for subclass sequential unlearning in Fig. 5.

Methods	**Hyperparameters**
Retrain	epoch $= 200$, lr $= 0.1$, milestones $= [60, 120, 160]$.
BT	epoch $= \{5, 10, 5, 10, 5\}$, lr $= \{0.5, 0.5, 1, 1, 5\} \times 10^{-5}$, temperature scalar $= \{5, 5, 5, 3, 1\}$.
FT	epoch $= 20$, lr $= 10^{-1}$ for all iterations.
SSD	dampening constant $\lambda = \{1, 0.1, 0.1, 1, 1\}$, selection weight $\alpha = \{71, 100, 90, 87, 85\}$.
SalUn	epoch $= 10$, lr $= 10^{-3}$, threshold $= 0.6$ for all iterations.
MU-Mis	epoch $= 30$, lr $= \{5, 1, 0.1, 3, 3\} \times 10^{-6}$, stopping threshold $\delta = \{1.4, 1.4, 0.95, 10, 10\}$.

G.4 KL DIVERGENCE

The KL divergence between two distributions is:

$$D_{\mathrm{KL}}(p_z(w_r)||p_z(w_u)) = \int p_z(w_r) \log[p_z(w_r)/p_z(\theta)] \mathrm{d}\mathcal{D} \tag{A3}$$

We calculate empirical KL divergence with the entire dataset (including both the train and valid set). We first collect the predicted class probabilities from both the unlearned and retrained models of each

sample, then we compute the output KL divergence as follows:

$$D_{\text{KL}} = \frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} p_c(x_i; w_r)\log\frac{p_c(x_i; w_r)}{p_c(x_i; w_u)}, \tag{A4}$$

where N is total number of dataset, C denotes the total number of classes, w_u is the unlearned model parameter and w_r is the retrained model parameter. $p_c(x_i, w)$ represents the c-th posterior probability of i-th sample in model w.

Table A4: Hyper-parameters range overview for different methods in all the experiments.

Methods	**Hyperparameters**
BT	epoch $\in \{1, 3, 5, 10\}$, lr $\in \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 5\times10^{-5}, 5\times10^{-6}, 10^{-6}\}$, temperature scalar $\in \{1, 3, 5\}$.
FT	epoch $\in \{5, 10, 15, 20\}$, lr $\in \{10^{-1}, 10^{-2}, 10^{-3}\}$.
SSD	dampening constant $\lambda \in \{0.1, 0.5, 0.9, 1\}$, selection weight $\alpha \in \{1, 5, 10, 20, 25, 30, 40, 50, 60, 70, 80, 90, 100\}$.
SCRUB	epoch $= 10$, lr $\in \{10^{-1}, 5\times10^{-2}, 10^{-2}, 5\times10^{-3}, 10^{-3}, 5\times10^{-4}, 10^{-4}, 5\times10^{-5}, 10^{-5}\}$, max step $\in \{2, 3, 5, 8\}$.
SalUn	epoch $\in \{10, 20\}$, lr $\in \{10^{-2}, 10^{-3}, 5\times10^{-4}, 5\times10^{-5}, 10^{-5}, 5\times10^{-6}, 10^{-6}\}$, threshold $\in \{0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$.
DUCK	lr $\in \{5\times10^{-2}, 10^{-2}, 5\times10^{-3}, 10^{-3}, 10^{-4}\}$, $\lambda_1, \lambda_2 \in \{0.5, 1, 1.2, 1.5, 2, 3, 5\}$.
NG	epoch $\in \{1, 3, 5, 10, 15, 20, 25, 30\}$, lr $\in \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$.
RL	epoch $\in \{1, 3, 5, 10, 15, 20, 25, 30\}$, lr $\in \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$.
JiT	dampening constant $= 1$, lr $\in [10^{-3}, 10^{-6}]$, lipschitz weight $\alpha \in [0, 1]$.
SCAR	lr $\in \{10^{-2}, 5\times10^{-3}, 10^{-3}, 5\times10^{-4}, 10^{-4}, 5\times10^{-5}, 5\times10^{-6}, 10^{-6}\}$, batch size $\in \{256, 512, 1024\}$, temperature $\in \{1, 3, 5\}$, $\lambda_1, \lambda_2 \in \{1, 1.5, 3, 5\}$.

H Additional Experiment Results

H.1 Unlearning Utility

H.1.1 Full class MU on PinsFaceRecognition

Results of full class unlearning on PinsFaceRecognition dataset are presented in Table A5. MU-Mis exhibits the smallest Avg.Gap to the retrained model, alongside a low MIA susceptibility. SCRUB and Salun exhibit a notably high MIA score, demonstrating a high risk of privacy leakage.

Table A5: Performance overview for **full class** unlearning (including MU-Mis and 6 baselines) evaluated on PinsFaceRecognition with ResNet-18. The content format follows Table 1.

Method	RA	FA	TA	Avg. Gap ↓	MIA	RTE
Pretrain	93.49	100	93.59	34.54	100.00	-
Retrain	$93.06_{\pm 0.21}$	$0.00_{\pm 0.00}$	$92.25_{\pm 0.32}$	0.00	$0.00_{\pm 0.00}$	-
BT	$92.69_{\pm 0.01}$	$0.00_{\pm 0.00}$	$91.48_{\pm 0.01}$	0.26	$0.00_{\pm 0.00}$	36
FT	$93.99_{\pm 0.09}$	$8.78_{\pm 1.34}$	$92.94_{\pm 0.10}$	3.47	$0.00_{\pm 0.00}$	146
SCRUB	$92.82_{\pm 0.02}$	$0.00_{\pm 0.00}$	$91.61_{\pm 0.02}$	0.29	$19.63_{\pm 0.28}$	112
SSD	$93.41_{\pm 0.00}$	$0.00_{\pm 0.00}$	$92.17_{\pm 0.00}$	0.14	$0.00_{\pm 0.00}$	8
DUCK	$92.17_{\pm 0.11}$	$0.00_{\pm 0.00}$	$91.06_{\pm 0.11}$	0.69	$0.00_{\pm 0.00}$	64
SalUn	$93.28_{\pm 0.05}$	$0.62_{\pm 0.00}$	$92.12_{\pm 0.06}$	0.34	$54.22_{\pm 1.06}$	154
MU-Mis	$92.98_{\pm 0.06}$	$0.00_{\pm 0.00}$	$92.13_{\pm 0.04}$	**0.07**	$0.00_{\pm 0.00}$	24

H.1.2 Sub class MU on CIFAR-20-Lamp

We evaluate sub-class unlearning on CIFAR-20-Lamp as presented in Table A6. The FA of the retrained model is 11.31, indicating certain generalization capability on the unlearned class. MU-Mis exhibits the smallest Avg.Gap to the retrained model.

Table A6: Performance overview for **sub-class** unlearning (including proposed MU-Mis and 6 baselines) evaluated on lamp of CIFAR-20 using ResNet-18. The content format follows Table 1.

Method	RA	FA	TA	Avg. Gap ↓	MIA	RTE
Pretrain	85.31	74.22	85.21	21.30	92.82	-
Retrain	$85.12_{\pm 0.22}$	$11.31_{\pm 1.60}$	$84.40_{\pm 0.20}$	0.00	$7.06_{\pm 0.11}$	-
BT	$85.52_{\pm 0.04}$	$10.00_{\pm 0.00}$	$84.84_{\pm 0.04}$	0.72	$0.00_{\pm 0.00}$	29
FT	$82.47_{\pm 0.15}$	$14.00_{\pm 2.19}$	$81.90_{\pm 0.18}$	1.97	$2.80_{\pm 0.36}$	128
SCRUB	$82.17_{\pm 0.68}$	$19.00_{\pm 4.74}$	$81.60_{\pm 0.70}$	4.48	$26.20_{\pm 4.18}$	113
SSD	$84.56_{\pm 0.00}$	$15.00_{\pm 0.00}$	$83.84_{\pm 0.00}$	1.60	$0.60_{\pm 0.00}$	18
DUCK	$83.25_{\pm 0.31}$	$31.02_{\pm 2.74}$	$82.69_{\pm 0.34}$	7.75	$27.68_{\pm 5.15}$	68
SalUn	$84.44_{\pm 0.05}$	$13.70_{\pm 1.66}$	$83.74_{\pm 0.04}$	1.24	$1.68_{\pm 0.17}$	1007
MU-Mis	$84.36_{\pm 0.51}$	$11.70_{\pm 1.29}$	$83.66_{\pm 0.50}$	**0.63**	$0.00_{\pm 0.00}$	10

H.1.3 Random subset unlearning

We evaluate random subset unlearning on CIFAR-10 and SVHN as presented in Table A6. MU-Mis exhibits consistently low MIA value, demonstrating good privacy preservation.

Table A7: Performance overview for **random subset (10%)** unlearning task evaluated on forgetting 10% CIFAR-10 and SVHN using ResNet-18. The content format follows Table 1.

Method	CIFAR-10						SVHN					
	RA	FA	TA	Avg. Gap↓	MIA	RTE	RA	FA	TA	Avg. Gap↓	MIA	RTE
Pretrain	100.00	99.96	94.68	1.84	92.64	-	100.00	100.00	96.48	1.65	83.11	-
Retrain	$100.00_{\pm 0.00}$	$94.51_{\pm 0.16}$	$94.75_{\pm 0.11}$	0.00	$84.77_{\pm 11.13}$	-	$100.00_{\pm 0.00}$	$95.12_{\pm 0.12}$	$96.40_{\pm 0.14}$	0.00	$81.24_{\pm 11.13}$	-
BT	$99.64_{\pm 0.01}$	$93.06_{\pm 0.06}$	$93.01_{\pm 0.04}$	1.18	$7.20_{\pm 0.00}$	50	$98.73_{\pm 0.01}$	$96.71_{\pm 0.03}$	$95.01_{\pm 0.01}$	1.42	$22.32_{\pm 0.00}$	215
FT	$100.0_{\pm 0.00}$	$94.73_{\pm 0.13}$	$93.17_{\pm 0.25}$	0.51	$68.66_{\pm 0.00}$	395	$100.0_{\pm 0.00}$	$96.38_{\pm 0.03}$	$96.87_{\pm 0.09}$	0.54	$80.72_{\pm 0.54}$	805
SCRUB	$100.00_{\pm 0.00}$	$95.79_{\pm 0.50}$	$93.43_{\pm 0.10}$	0.87	$77.56_{\pm 0.93}$	207	$95.40_{\pm 1.18}$	$94.79_{\pm 1.00}$	$94.94_{\pm 0.83}$	1.98	$23.15_{\pm 2.62}$	104
SSD	$99.99_{\pm 0.00}$	$99.98_{\pm 0.02}$	$94.66_{\pm 0.01}$	1.86	$92.54_{\pm 0.00}$	18	$98.67_{\pm 0.39}$	$98.63_{\pm 0.43}$	$96.59_{\pm 0.11}$	1.52	$84.23_{\pm 0.77}$	55
DUCK	$98.04_{\pm 0.05}$	$97.90_{\pm 0.21}$	$92.38_{\pm 0.06}$	2.57	$90.59_{\pm 0.29}$	21	$96.16_{\pm 0.20}$	$94.86_{\pm 0.39}$	$95.53_{\pm 0.25}$	1.51	$30.46_{\pm 9.28}$	156
SalUn	$100.0_{\pm 0.00}$	$94.31_{\pm 0.03}$	$93.19_{\pm 0.03}$	**0.59**	$26.52_{\pm 0.00}$	247	$99.77_{\pm 0.00}$	$94.50_{\pm 0.05}$	$95.85_{\pm 0.01}$	**0.47**	$19.92_{\pm 0.00}$	409
NG	$96.46_{\pm 0.23}$	$96.15_{\pm 0.35}$	$90.52_{\pm 0.22}$	3.13	$88.21_{\pm 0.32}$	25	$98.98_{\pm 0.02}$	$98.98_{\pm 0.05}$	$96.56_{\pm 0.01}$	1.53	$84.53_{\pm 0.26}$	25
RL	$96.17_{\pm 0.28}$	$93.99_{\pm 0.51}$	$88.29_{\pm 0.34}$	4.27	$83.11_{\pm 0.50}$	26	$98.66_{\pm 0.08}$	$98.53_{\pm 0.10}$	$96.02_{\pm 0.02}$	1.56	$73.94_{\pm 0.28}$	25
JiT	$95.45_{\pm 1.92}$	$95.46_{\pm 1.81}$	$89.63_{\pm 1.90}$	3.54	$86.00_{\pm 0.32}$	255	$96.30_{\pm 0.64}$	$96.26_{\pm 0.74}$	$95.14_{\pm 0.89}$	1.88	$62.78_{\pm 24.32}$	333
SCAR	$98.63_{\pm 0.13}$	$98.64_{\pm 0.08}$	$92.43_{\pm 0.13}$	2.61	$48.36_{\pm 0.35}$	197	$96.34_{\pm 0.54}$	$96.46_{\pm 0.69}$	$92.39_{\pm 0.70}$	2.86	$55.20_{\pm 2.86}$	158
MU-Mis	$97.76_{\pm 0.03}$	$97.43_{\pm 0.04}$	$91.50_{\pm 0.03}$	2.80	$33.04_{\pm 0.28}$	116	$95.48_{\pm 0.00}$	$95.50_{\pm 0.03}$	$94.22_{\pm 0.00}$	2.36	$26.11_{\pm 0.00}$	116

H.1.4 FULL CLASS MU COMPARED WITH JIT WITH VGG16

JiT unlearns by regularizing lipshitz constant around the forgetting data, which might fail on DNNs with batch norm layers. Following the architecture used in the original paper, we compare with it on CIFAR-100-rocket unlearning in Table A8. The results show that JiT exhibits greater efficacy when implemented with VGG-16 as opposed to ResNet-18. However, it still exhibits a noticeable performance disparity when compared to MU-Mis.

Table A8: Performance comparison between MU-Mis and JiT of **full class** unlearning on CIFAR-100 using **VGG-16**.

Metrics	RA	FA	TA	Avg. Gap↓	MIA	RTE
Pretrain	64.72	66.75	64.76	23.64	85.20	-
Retrain	$64.96_{\pm 0.35}$	$0.00_{\pm 0.00}$	$63.36_{\pm 0.34}$	0.00	$15.7_{\pm 1.56}$	-
JiT	$60.98_{\pm 0.08}$	$3.73_{\pm 0.00}$	$60.45_{\pm 0.07}$	3.21	$8.02_{\pm 0.24}$	16
MU-Mis	$64.60_{\pm 0.04}$	$0.00_{\pm 0.28}$	$63.96_{\pm 0.04}$	**0.32**	$3.28_{\pm 0.22}$	35

H.1.5 SUB-CLASS MU ON CIFAR-20-SEA WITH VIT

We conducted experiments with ViT on subclass-CIFAR20-sea to further demonstrate our effectiveness across different tasks. MU-Mis exhibits the best RA, FA, and TA, and provides advantages in unlearning time.

Table A9: Performance overview for **sub-class** unlearning task evaluated on **Cifar20-Sea** using **ViT**. RTE is reported in **minute**.

Methods	RA	FA	TA	Avg. Gap ↓	MIA	RTE (min)
Pretrain	93.65	91.32	93.63	0.93	69.80	-
Retrain	$93.90_{\pm 0.12}$	$88.98_{\pm 0.08}$	$93.84_{\pm 0.14}$	0.00	$59.00_{\pm 0.23}$	-
SalUn	$94.15_{\pm 0.10}$	$89.29_{\pm 0.03}$	$94.13_{\pm 0.08}$	0.27	$62.10_{\pm 0.42}$	10
MU-Mis	$93.70_{\pm 0.02}$	$88.84_{\pm 0.25}$	$93.67_{\pm 0.02}$	**0.17**	$69.67_{\pm 0.31}$	0.5

H.2 ABLATION STUDY

We study the role of each term in our loss through ablation, illustrating with fullclass-CIFAR100-Rocket on ResNet-18. We denote the first term $\|\nabla_x f_c(w,x)\|_F^2$ in our loss Eq. (3) as TC (Target Class) and the second term $\|\nabla_x f_{c'}(w,x)\|_F^2$ as OC (Other Class). In Table A10, we showcase the unlearning performance of decreasing TC, increasing OC, and decreasing TC - OC (MU-Mis) respectively. From the ablation study, we observe that minimizing the first term obliviates information of the forgetting data, but greatly hurts performance on the remaining data. Solely increasing OC brings a slight accuracy drop on remaining data, but hardly unlearns the forgetting data. While our MU-Mis loss could unlearn effectively meanwhile preserve model utility on the remaining data.

Table A10: Ablation study on each term of our loss in full class (Rocket) unlearning. TC (Target Class) refers to the first term and OC (Other Class) refers to the second term.

Methods	RA	FA	TA	Avg. Gap ↓	MIA	RTE
Pretrain	76.41	79.69	76.47	26.84	95.80	-
Retrain	$76.52_{\pm 0.27}$	$0.00_{\pm 0.00}$	$75.76_{\pm 0.24}$	0.00	$2.87_{\pm 0.46}$	-
TC	$65.41_{\pm 0.00}$	$0.00_{\pm 0.00}$	$64.73_{\pm 0.00}$	7.38	$1.40_{\pm 0.00}$	44
OC	$70.13_{\pm 0.41}$	$74.62_{\pm 1.56}$	$70.21_{\pm 0.42}$	28.85	$0.00_{\pm 0.00}$	25
TC - OC	$76.42_{\pm 0.07}$	$0.00_{\pm 0.00}$	$75.64_{\pm 0.07}$	**0.07**	$0.00_{\pm 0.00}$	30

H.3 HYPER-PARAMETER SENSITIVITY

We show that there is a clear relationship between the norm ratio and model performance during unlearning in Figure A1. The stopping threshold ratio δ controls the magnitude ratio of final irrelevant class sensitivity norm $\|\nabla_x f_{c'}(x,w)\|_F$ to initial one $\|\nabla_x f_{c'}(x,w_p)\|_F$. In this part, we investigate the sensitivity of unlearning performance to δ in Figure A2, taking fullclass-CIFAR100-Rocket with ResNet-18 as an example. Different colors represent different δ, ranging from 70% to 100%. The green dashed line indicates the performance of the retrained model. Notably, under a specific learning rate, the Avg. Gap exhibits minimal variation with changes in δ. Generally, MU-Mis demonstrates resilience against variations in hyper-parameters δ. In summary, the hyper-parameter tuning for MU-Mis is effortless, resilient and well-guided. This advantage in hyper-tuning indirectly enhances the efficiency of unlearning and facilitates straightforward application of MU-Mis, making it valuable for practical applications of MU methods.

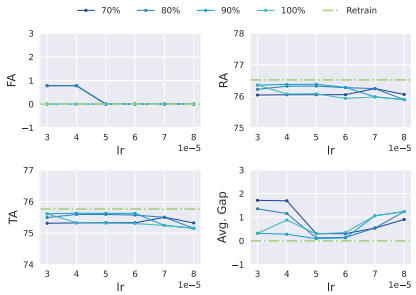

Figure A2: MU-Mis unlearning performance with different stopping threshold ratio under different learning rates on fullclass-CIFAR100-rocket with ResNet-18. Performance varies little with the threshold ratio when the learning rate is fixed.

H.4 Effectiveness Visualization by Attention Map

To further investigate and understand the behavior of our unlearned model, we showcase the attention heatmaps (Selvaraju et al., 2017) of models before and after applying MU-Mis on PinsFaceRecognition dataset in Figure A3. For the forgetting data, the original attention concentrates on the faces. After applying MU-Mis, the attention on the faces either disappears or significantly weakens, and is shifted towards the background. For the remaining data, MU-Mis fully maintains previous attention. Notably, an alternative interpretation of input sensitivity is the measurement of how changes in the image influence its model prediction (Smilkov et al., 2017). Our proposed method reduces the target class logit sensitivity to the forgetting data while recovering irrelevant classes', thereby enabling the unlearned model to disregard the semantic information in the forgetting data meanwhile preserve prediction sensitivity and performance on the remaining data.

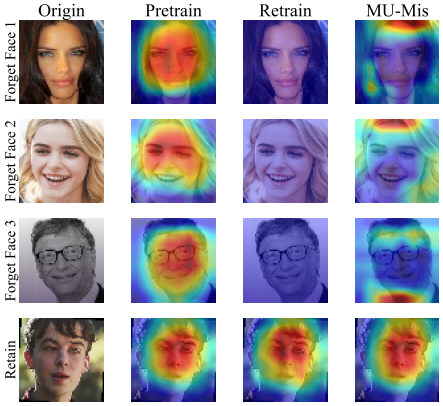

Figure A3: **Visualization of attention maps for the full class unlearning task on PinsFaceRecognition.** MU-Mis distracts attention from forgetting data regions while preserving attention on remaining data.

I Underlying Reasons for Model Utility Preservation

We suggest the favorable preservation on model utility might be attributed to two factors:

- **Conceptual motivations**: The forgetting operation in MU-Mis differs fundamentally from previous methods. While previous methods involved relabelling the forgetting data (Liu et al., 2024; Foster et al., 2024a; Fan et al., 2024) or knowledge distillation from a useless teacher (Chundawat et al., 2023; Lin et al., 2023; Kurmanji et al., 2024), which unlearn by introducing incorrect information to spoil original knowledge, MU-Mis unlearns by solely withdrawing the contribution of forgetting data. In Figure A1, we show that RA and TA stay at a high level throughout the withdrawal process.
- **Empirical investigation**: Current gradient-based unlearning methods (Liu et al., 2024; Foster et al., 2024a; Fan et al., 2024; Wu et al., 2020; Graves et al., 2021; Neel et al., 2021; Thudi et al., 2022) all employ cross-entropy loss gradient $\nabla_\theta \mathcal{L}(w, x)$ to unlearn. However, the intra-class gradients in a well-trained model are quite similar (Papyan, 2020; Papyan et al., 2020). While the gradient of input sensitivity norm w.r.t parameters is double back-propagation (Drucker & Le Cun, 1992), enhancing sample-specificity by first back-propagating to the input samples before reaching the parameters. Our pairwise analysis of cosine similarity among intra-class and inter-class samples, detailed in the following, reveals a distinctive orthogonality in input sensitivity gradients, spontaneously reducing the interference between the forgetting and remaining data.

In the following part, we empirically demonstrate an inherent orthogonality among samples from the perspective of input sensitivity.

We calculate the pairwise cosine similarity within a class and between classes of the derivatives of four metrics w.r.t. parameters in a well-trained model. They are $\nabla_w \mathcal{L}$, $\nabla_w f_c$, $\nabla_w \|\nabla_x f_c\|_F$, and $\nabla_w \sum_{c' \neq c} \|\nabla_x f_{c'}\|_F$ (denoted as $\nabla_w \|\nabla_x f_{c'}\|_F$ in the following for brevity). The first two metrics are commonly used in current unlearning methods, and the last two are utilized in MU-Mis. The pair-wise similarity results are shown in Figure A4. Derivatives of all four metrics are approximately orthogonal between samples from different classes. However, intra-class similarities differ across four metrics. We can see that both $\nabla_w \mathcal{L}$ and $\nabla_w f_c$ bear a resemblance within a class, with cosine similarity centering around 0.3 and reaching up to 0.6. On the other hand, $\nabla_w \|\nabla_x f_c\|_F$ centers around 0.1 with intra-class similarity scarcely exceeding 0.3. Directly taking the derivative of output w.r.t parameters preserves within-class similarity of the output, but such similarity is reduced when output first takes the derivative to the input and then back to parameters.

Interestingly, $\nabla_w \sum_{c' \neq c} \|\nabla_x f_{c'}\|_F$ seems to be pairwise similar. However, we speculate that there are two reasons why this portion of our loss function does not significantly harm the performance of the remaining data. Firstly, as shown in Figure 3, $\sum_{c' \neq c} \|\nabla_x f_{c'}\|_F$ is significantly smaller than $\|\nabla_x f_c\|_F$ on well-trained models. Therefore, in the early stages of optimizing the relative magnitudes of input sensitivities, the contribution of $\nabla_w \sum_{c' \neq c} \|\nabla_x f_{c'}\|_F$ to the optimization direction can be neglected. Secondly, $\sum_{c' \neq c} \|\nabla_x f_{c'}\|_F$ for the forgetting data actually corresponds to the target class of the remaining data. The inter-class similarity of $\nabla_w \sum_{c' \neq c} \|\nabla_x f_{c'}\|_F$ implies that when we increase the input sensitivity of the forgetting data for the other irrelevant classes, we also increase the input sensitivity of the remaining data of the corresponding classes, thus preserving their sample contributions.

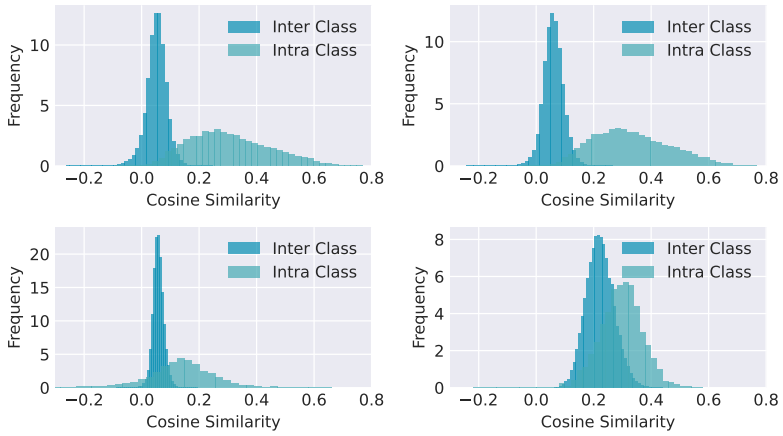

Figure A4: **Inter-class and intra-class cosine similarity of four different metrics w.r.t parameters.** From left to right: $\nabla_w \mathcal{L}$, $\nabla_w f_c$, $\nabla_w \|\nabla_x f_c\|_F$, and $\nabla_w \|\nabla_x f_{c'}\|_F$. Directly taking the derivative of output w.r.t parameters bears a resemblance across samples, but such similarity is reduced when output first takes derivative to input and then back to parameters.

J Discussion

Beyond the advantages of input sensitivity for measuring sample contribution, we raise the attention that the input sensitivity perspective possesses a profound connection to MU. MU emerges from model's memorization of the training data and seeks to safeguard the privacy of training data. Recent studies by Garg et al. (Garg et al., 2024) and Ravikumar et al. (Ravikumar et al., 2024) reveal the intrinsic relationship between memorization, privacy and sample's input curvature. Also, Mo et al. (Mo et al., 2021) demonstrates that the input sensitivity of model gradient is the underlying cause of information leakage exposed by Model Inversion Attack (MIA) (Zhu et al., 2019; Zhao et al., 2020). Collectively, these works further underscore the inherent advantages of adopting an input sensitivity perspective for machine unlearning.