Forget Vectors at Play: Universal Input Perturbations Driving Machine Unlearning in Image Classification

Anonymous Submission

Abstract

Machine unlearning (MU), which seeks to erase the influence of specific unwanted data from already-trained models, is becoming increasingly vital in model editing, particularly to comply with evolving data regulations like the "right to be forgotten". Conventional approaches are predominantly model-based, typically requiring retraining or fine-tuning the model's weights to meet unlearning requirements. In this work, we approach the MU problem from an input perturbation-based perspective, where the model weights remain intact throughout the unlearning process. We demonstrate the existence of a proactive input-based unlearning strategy, referred to *forget vector*, which can be generated as an input-agnostic data perturbation and remains as effective as model-based approximate unlearning approaches. We also explore *forget vector arithmetic*, whereby multiple class-specific forget vectors can be combined through simple operations (*e.g.*, linear combinations) to generate new forget vectors for unseen unlearning tasks, such as forgetting arbitrary subsets across classes. Extensive experiments validate the effectiveness and adaptability of the forget vector, showcasing its competitive performance relative to state-of-the-art model-based methods while achieving superior parameter efficiency.

1 Introduction

To prevent unauthorized use of personal or sensitive data after training and comply with legislation such as the "right to be forgotten" [1], machine unlearning (**MU**) has garnered increasing attention as a solution to various challenges in vision tasks [2–5]. In essence, it initiates a reverse learning process to erase the impact of unwanted data (*e.g.*, specific data points, classes, or knowledge) from an already-trained model, while still preserving its utility for information not targeted by an unlearning request. Based on the guarantees provided for data removal from already-trained models, existing MU methods can be broadly categorized into two approaches: *exact unlearning* [6–8] and *approximate unlearning* [9–15]. The former guarantees the complete and verifiable removal of targeted data, typically achieved by retraining the model from scratch with the data to be forgotten excluded from the training set, a process we refer to as *Retrain*. However, due to the high computational overhead, research has increasingly focused on approximate unlearning methods, which seek to achieve efficient unlearning without requiring full retraining.

Approximate unlearning strikes a balance between computational efficiency and effective data removal, making it practical for many real-world applications. Most existing approximate unlearning techniques are *model-based*, updating the model's weights within a limited number of training iterations to eliminate the influence of specific unwanted data, thus avoiding a full retraining process. Representative methods in this category include fine-tuning approaches [4, 16], gradient ascent techniques [10, 17], and influence function-based methods [18, 2].

Although the model-based unlearning methods have made significant strides, they often overlook the *data-based* dimension and its potential impact on MU. For instance, it remains unclear whether current MU approaches generalize effectively to "shifted" forget data. Additionally, the possibility

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

of a data-based MU design that operates without updating model parameters has yet to be explored. Therefore, we ask:

(Q) *Can we explore data influence in MU and harness data-based operations to fulfill MU?*

To address **(Q)**, we study MU from a fresh data-based viewpoint: **forget vector**, a universal input data perturbation designed to promote unlearning effectively; See the schematic overview in **Fig. 1**. Before developing the forget vector, we explore the rationale for how data perturbations complement current model-based MU approaches, as evidenced by these methods' generalization to common data shifts, including Gaussian noise and adversarial perturbations [20, 21]. To design the forget vector, we draw inspiration from recent input prompting techniques for vision models, known as visual prompting [22–24] or model reprogramming [25–27], used in transfer learning and model adaptation. These prompting methods learn input perturbations to enable a fixed model to perform well on new tasks, effectively guiding the model to execute tasks it wasn't originally trained for. From this perspective, our research on the forget vector also explores whether it is possible to append a trainable "prompt" to the input to guide an already-trained neural network in unlearning specific data. The proposed forget vector allows the unlearner to modify user inputs targeted for deletion, offering a flexible and efficient approach to unlearning while potentially achieving significant parameter efficiency. We summarize **our contributions** below.

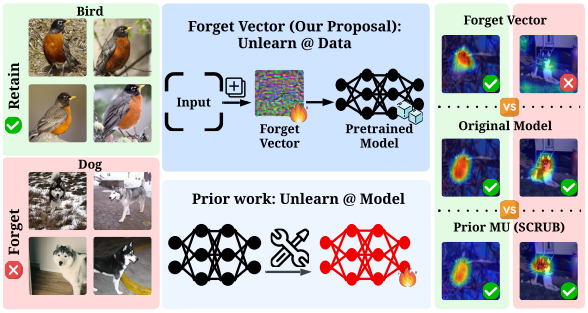


Figure 1: A schematic illustration comparing our proposed data-based MU method (termed the "forget vector"), which achieves unlearning objectives (*i.e.*, forgetting "dog" and remembering "bird" in this example) by operating directly on input data without altering model parameters, against traditional model update-based unlearning methods. ✗ indicates that the forget data is successfully unlearned, while ✓ means that the retain data is correctly recognized, or the forget data is *not* successfully unlearned. The "original model" refers to the model without unlearning applied, and "SCRUB" [19] is an existing representative unlearning method that updates model weights.

- We investigate the impact of forget data shifts on image classifiers post-unlearning, revealing that unlearning demonstrates resilience against these shifts (to some extent) while generalization remains more vulnerable.
- Building on the complementary role of data shifts in MU, we propose a proactive, input-agnostic data perturbation strategy termed the *forget vector*, optimized specifically to facilitate unlearning.
- We demonstrate the effectiveness of *forget vector arithmetic* by using precomputed *class-wise* forget vectors to generate new vectors that effectively eliminate the influence of specific data subsets in image classification models, *e.g.*, in the scenario of *random* data forgetting.
- We conduct extensive experiments on MU for image classification, providing both quantitative and qualitative analyses to demonstrate the competitiveness of the forget vector compared to model-based MU methods.

2 Related Work

MU in Vision. Machine unlearning (MU) in vision has gained significant attention due to the increasing need for privacy preservation, copyright protection, and ethical data removal in machine learning models. Recent studies [28–33, 19, 34–37] in this area have primarily focused on two main applications: image classification and image generation.

In **image classification**, MU methods have explored various ways to erase certain classes or images from models [10, 17, 2, 18, 37, 38]. Specifically, fine-tuning-based methods update the model incrementally on a modified dataset without the unwanted data points [4, 16]. Gradient ascent-based approaches attempt to reverse the impact of unwanted data by applying gradient ascent to model parameters [10, 17]. Influence-based unlearning techniques estimate and negate the effect of specific data points on model predictions and parameters to achieve unlearning [2, 18]. Another line of research explores the relationship between MU and model pruning, suggesting that model sparsity

can help to bridge the gap between approximate and exact unlearning, reducing the need for complex parameter updates [37, 38].

For **image generation**, MU techniques [5, 30, 36] have been proposed to prevent models from generating unwanted or harmful content while retaining high-quality outputs. For example, weight saliency methods guide MU by identifying and selectively altering model parameters to eliminate specific content generation [5]. Beyond vision, MU has been applied to other domains, with notable efforts in natural language processing [39–42], graph-based data [43, 6], and time-series data [44]. However, most existing MU methods are model-based, requiring updates to model parameters and consequently incurring high computational costs.

Input-based Model Adaptation. This approach aims to modify or repurpose pre-trained models for new tasks or specific objectives without the need for full retraining. It is particularly beneficial for reducing computational costs and leveraging existing knowledge within models. Key techniques in model adaptation include: Visual prompting [45–49] maintains the pre-trained model's parameters fixed and adapts the input to enable the model to perform different tasks. For example, introducing trainable parameters in the input space while keeping the model backbone frozen can achieve comparable results with reduced computational overhead. Model reprogramming [50, 25, 51, 52] involves keeping a pre-trained model unchanged while modifying its inputs to adapt the model for new tasks. For example, adversarial perturbations can be applied to inputs at test time, allowing the model to perform a specific task dictated by the perturbations, even if that task was not originally intended for the model. Feature-based domain adaptation [53, 54] applies transformations or mapping techniques to the input data, aligning the feature distributions between the source and target domains while keeping the model unchanged.

3 Preliminaries on MU and Problem Statement

In this section, we introduce the fundamentals of machine unlearning (MU), including its formulation, commonly-used methods, evaluation metrics, and motivate our focus: a data-based forget vector design for achieving MU.

Formulation of MU. In this work, we focus on the problem of MU for image classification. Let $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$ represent a training set with N examples, where $\mathbf{x}_i$ denotes the ith image data, and y_i denotes its corresponding class label. Following the classic MU setting [2, 55, 19, 5], we introduce a *forget set* $\mathcal{D}_\mathrm{f} \subseteq \mathcal{D}$, which specifies the training samples targeted for unlearning. Accordingly, the complement of $\mathcal{D}_\mathrm{f}$ is the *retain set*, *i.e.*, $\mathcal{D}_\mathrm{r} = \mathcal{D} \setminus \mathcal{D}_\mathrm{f}$. The *goal of MU* is to efficiently and effectively eliminate the influence of $\mathcal{D}_\mathrm{f}$ on an already-trained model $\boldsymbol{\theta}_\mathrm{o}$, so that the performance of the post-unlearning model closely approximates that of a model retrained from scratch on $\mathcal{D}_\mathrm{r}$ (*i.e.*, excluding the impact of $\mathcal{D}_\mathrm{f}$ from scratch). Therefore, such a retraining method (referred to as *Retrain*) is typically considered as the gold standard of MU [10, 56]. However, since Retrain is computationally intensive, most popular MU approaches instead address an unlearning optimization problem using the forget and retain sets to update the model parameters $\boldsymbol{\theta}$, starting from the originally pre-trained model $\boldsymbol{\theta}_\mathrm{o}$. This yields the following optimization problem for MU:

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad \ell_\mathrm{MU}(\boldsymbol{\theta}; \mathcal{D}_\mathrm{f}, \mathcal{D}_\mathrm{r}), \tag{1}$$

with the initialization $\boldsymbol{\theta} = \boldsymbol{\theta}_\mathrm{o}$. In (1), ℓ_MU represents an appropriate unlearning loss function that may depend on $\mathcal{D}_\mathrm{f}$ and/or $\mathcal{D}_\mathrm{r}$, as will be detailed when introducing specific unlearning methods. In the context of MU for image classification [2, 5], the specification of the forget set $\mathcal{D}_\mathrm{f}$ leads to two unlearning scenarios: *class-wise forgetting*, where $\mathcal{D}_\mathrm{f}$ consists of a subset focused on a specific image class targeted for unlearning, and *random data forgetting*, where $\mathcal{D}_\mathrm{f}$ is a randomly selected subset of images across all classes.

Model-based MU Methods and Evaluation. The formulation in (1) represents the predominant MU solution in the literature, focusing on modifying model weights and/or architectural components to achieve the unlearning objective.

In what follows, we introduce several representative MU approaches that serve as approximations to Retrain. (a) Fine-tuning **(FT)** [4]: This approach treats the MU problem as a continual learning task, defining the unlearning objective ℓ_MU as a training objective that fine-tunes $\boldsymbol{\theta}_\mathrm{o}$ over $\mathcal{D}_\mathrm{r}$ to induce catastrophic forgetting of $\mathcal{D}_\mathrm{f}$. (b) Random labeling **(RL)** [2]: This approach specifies the unlearning objective ℓ_MU by assigning random labels or features to the data in $\mathcal{D}_\mathrm{f}$, thereby enforcing model forgetting. (c) Gradient ascent **(GA)** [10]: This approach employs the negative of the FT loss to

reverse the training impact associated with the data in $\mathcal{D}_\mathrm{f}$. (d) **Localization**-informed unlearning [56, 5]: This method identifies a subset of model weights critical to the unlearning task (*e.g.*, through model sparsity [56] or gradient saliency [5]) and incorporates this weight localization as a prior to solve the unlearning problem in (1).

Given an unlearned model (denoted as $\boldsymbol{\theta}_\mathrm{u}$) after solving (1), unlearning performance is evaluated in two main areas: *unlearning effectiveness*, which measures whether the target data/information has been successfully removed, and *utility retention*, which assesses whether unlearning has preserved the model's classification ability on unaffected data. Following the evaluation pipeline in [56], unlearning effectiveness is quantified by two metrics: unlearning accuracy (**UA**), defined as $1-$the model's accuracy on $\mathcal{D}_\mathrm{f}$ (higher UA indicates better unlearning), and membership inference attack performance on $\mathcal{D}_\mathrm{f}$, termed **MIA-Efficacy**, where higher prediction accuracy on non-training samples indicates better unlearning (see Appendix A). Utility retention is measured by retain accuracy (**RA**), reflecting the model's accuracy on $\mathcal{D}_\mathrm{r}$, and testing accuracy (**TA**), which is the accuracy on the original test set. Notably, TA is assessed on the entire original test set, except in the case of class-wise forgetting, where test samples from the forgotten class are excluded from evaluation.

Data-based MU Design: The Forget Vector Problem. While previous MU methods can be unified within the framework of (1) by varying the unlearning loss ℓ_MU and weight localization priors, recent advancements in input data-based model adaptation, such as visual prompting [57, 23] and model reprogramming [27, 25], suggest an alternative approach to MU. This strategy inspires us to design data-based prompting (implemented through universal input perturbations) to achieve unlearning without modifying the model itself. We refer to this input perturbation vector, designed specifically for MU, as the **forget vector**. To be more specific, let $\boldsymbol{\delta}$ represent the data-agnostic input perturbations to be designed. The problem of constructing a forget vector for MU can be formulated as

$$\underset{\boldsymbol{\delta}}{\text{minimize}} \quad \ell_\mathrm{MU}(\boldsymbol{\delta}; \boldsymbol{\theta}_\mathrm{o}, \mathcal{D}_\mathrm{f}, \mathcal{D}_\mathrm{r}), \tag{2}$$

where $\boldsymbol{\delta}$ is the perturbation variable, applied linearly to the forget and retain samples as $\mathbf{x}' := \mathbf{x} + \boldsymbol{\delta}$ for $\mathbf{x} \in \mathcal{D}_\mathrm{f}$ and $\mathcal{D}_\mathrm{r}$, similar to visual prompting [57] and adversarial examples [21]. In practice, since the model remains unchanged, the unlearner can compute the forget vector based on the forget request (forget set) and append it to model inputs to process user-initiated unlearning requests. In this work, we do not consider counter-unlearning adversaries that intentionally negate the effect of the forget vector. We will detail the unlearning objective function required for designing the forget vector in our later method sections.

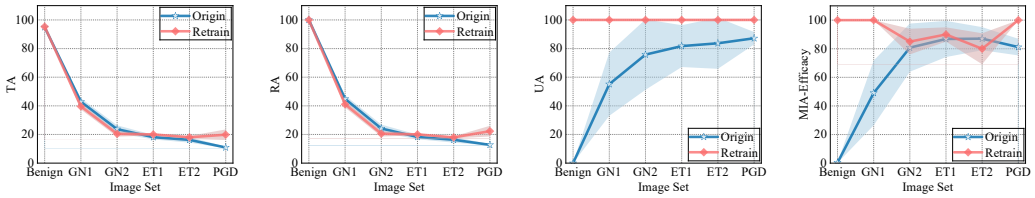
Based on (2), we are motivated to explore two research questions: **(Q1)** How do "perturbations" applied to forget data affect unlearning performance? **(Q2)** How can we effectively design the forget vector $\boldsymbol{\delta}$ to solve problem (2)? These two questions are interconnected: the answer to (Q1) offers a sensitivity analysis of MU to data shifts within the forget set, guiding how the specific shift induced by the forget vector can be optimized for effective unlearning in (Q2). Therefore, the following Secs. 4-5 address (Q1) and (Q2) in sequence. For (Q1), the next section analyzes performance through an evaluation lens on a given unlearned model, using data perturbations applied via standard data augmentation operations or adversarial perturbations.

4 Generalization of MU to Forget Data Shifts

Before designing the forget vector as formulated in (2), we examine the sensitivity of existing model-based unlearning approaches to external perturbations applied to forget data. Such a perturbation-based or out-of-distribution (OOD) generalization analysis of MU has not been explored in the literature. Our rationale is that if conventional MU approaches demonstrate robustness to these external forget data perturbations post unlearning, then enhancing MU with a forget vector could become a seamless process, as a proactive design of such a vector would likely yield effective results.

Post-unlearning Forget Data Perturbations. Given an unlearned model ($\boldsymbol{\theta}_\mathrm{u}$) after solving (1), we examine two types of shifts in forget data: standard data corruptions used in the evaluation of OOD generalization [58, 20] and worst-case perturbations generated by adversarial attacks [21, 59].

(a) Data Corruptions. Following the OOD generalization evaluation approach in image classification [58, 20], we consider data corruptions from four main categories: noise, blur, weather, and digital. Each type of corruption includes five levels of severity, with higher levels representing increased noise intensity. Among these, we select zero-mean Gaussian noise (GN) and Elastic transformations (ET) as the primary corruption types to evaluate MU robustness against shifts in forget data. Our rationale



(a) TA (b) RA (c) UA (d) MIA-Efficacy

Figure 2: The performance of class-wise forgetting on (ResNet-18, CIFAR-10) using the unlearning method *Retrain* vs. the (pre-unlearning) original model performance (Origin), evaluated on both benign evaluation sets (Benign) and perturbed sets, which include (1) Gaussian noise (GN) with a standard deviation of 0.08 (termed GN1), (2) GN with a standard deviation of 0.2 (termed GN2), (3) Elastic transformation (ET) with parameters (488, 170.8, 24.4) regarding intensity, smoothing, and offset (termed ET1), (4) ET with parameters (488, 19.52, 48.8) (termed ET2), and (5) adversarial perturbations from a 7-step PGD attack with strength $\epsilon = 8/255$. The unlearning performance metrics are reported as (a) TA (testing accuracy), (b) RA (retain accuracy), (c) UA (unlearning accuracy), and (d) MIA-Efficacy, as defined in Sec. 3. The average performance is reported over 10 independent trials, where each trial focuses on forgetting one specific class from CIFAR-10. Shaded regions indicate the performance variance.

is that Gaussian noise yields small pixel-wise perturbations (similar to adversarial perturbations introduced later) and Elastic transformations stretch or contract small image regions.

(b) Adversarial perturbations. An adversarial image is a benign image altered with carefully crafted, pixel-wise perturbations designed to mislead a classifier. In this work, we use the ϵ-constrained ℓ_∞ norm-based K-step projected gradient descent (PGD) attack [21, 59] to generate adversarial examples via iterative projected gradient updates. The parameter $\epsilon > 0$ defines the radius of the ℓ_∞ norm of the perturbations, controlling their strength. And K represents the number of PGD steps.

Generalization of MU to Forget Data Perturbations. Next, we apply the above data shift operations to the MU evaluation sets–namely, the forget, retain, and testing sets–and assess the unlearning performance of an unlearned model. **Fig. 2** displays the performance of the gold standard unlearning method, Retrain, against Gaussian noise at test time with standard deviations of 0.08 and 0.2 [20], and two types of Elastic transformations with parameters $(488, 170.8, 24.4)$ and $(488, 19.52, 48.8)$ regarding intensity, smoothing and offset for moderate and high-intensity distortions [20], as well as a 7-step PGD attack with perturbation strength $\epsilon = 8/255$ [21]. To ensure the feasibility of Retrain, we conduct the image classification task using ResNet-18 on the CIFAR-10 dataset.

As shown in Fig. 2-(a) and (b), model utility, measured by RA (retain accuracy) and TA (testing accuracy), decreases when external perturbations are applied to the evaluation sets compared to its original performance without perturbations. This is expected due to the generalization loss when evaluated on new, shifted data. More interestingly, Fig. 2-(c) and (d) show that unlearning effectiveness of Retrain, measured by UA (unlearning accuracy) and MIA-Efficacy, remains stable despite the presence of these perturbations on the forget set. This is because perturbations degrade prediction performance across evaluation sets, including the forget set. This is further evidenced by the increase in UA and MIA-Efficacy for the original model (without unlearning) when exposed to data perturbations. The above indicates that a reduction in prediction performance on the forget set could translate into enhanced unlearning effectiveness on that set. In Appendix B, we provide additional evaluations of other approximate unlearning methods, including FT, RL, and GA, showing consistent performance.

The results above demonstrate that unlearning effectiveness is inherently preserved under external perturbations at no additional cost. However, balancing this with utility retention in the presence of perturbations remains challenging and desirable. Therefore, we need to carefully address the forget vector problem (2) to develop an input-based MU solution that enhances unlearning effectiveness without compromising model utility.

5 Optimization for Forget Vectors

In this section, we first propose an unlearning objective function, ℓ_{MU}, tailored for the forget vector design in (2), inspired by the untargeted C&W attack [60]. We then introduce a novel paradigm called compositional unlearning, facilitated by forget vector arithmetic.

Unlearning Objective Design of Forget Vectors. Our design aims for the forget vector variable ($\boldsymbol{\delta}$), when applied to the forget set ($\mathcal{D}_{\mathrm{f}}$), to drive the given model's predictions ($\boldsymbol{\theta}_{\mathrm{o}}$) away from the correct labels. Conversely, when applied to the retain set ($\mathcal{D}_{\mathrm{r}}$), the forget vector should minimally

affect correct predictions. The first forget objective aligns with adversarial attack design, aiming to mislead the model's predictions in the presence of the perturbation $\boldsymbol{\delta}$. The second retain objective acts as a utility regularization, suppressing the unlearning effect of the perturbation when applied to data samples not targeted for unlearning (retain samples).

To implement the forget objective (denoted by ℓ_{f}), we draw inspiration from the C&W untargeted attack loss [60]. This is given by a margin loss, designed to remain actively minimizing when the top prediction matches the correct label, ensuring that optimization continues until predictions are shifted to an incorrect label, thereby achieving unlearning. This can be cast as

$$\ell_{\mathrm{f}}(\boldsymbol{\delta}; \boldsymbol{\theta}_{\mathrm{o}}, \mathcal{D}_{\mathrm{f}}) = \mathbb{E}_{(\mathbf{x},y)\in\mathcal{D}_{\mathrm{f}}} \max\{f_{\boldsymbol{\theta}_{\mathrm{o}},y}(\mathbf{x}+\boldsymbol{\delta}) - \max_{k:\, k\neq y} f_{\boldsymbol{\theta}_{\mathrm{o}},k}(\mathbf{x}+\boldsymbol{\delta}), -\tau\}, \tag{3}$$

where $(\mathbf{x}, y) \in \mathcal{D}_{\mathrm{f}}$ denotes a forget sample with y being the prediction label of $\mathbf{x}$, $\mathbf{x}+\boldsymbol{\delta}$ is the perturbed sample, $f_{k,\boldsymbol{\theta}_{\mathrm{o}}}(\mathbf{x})$ denotes the prediction logit (before softmax) of the model $\boldsymbol{\theta}_{\mathrm{o}}$ for class k under the input $\mathbf{x}$, and $\tau \geq 0$ is a margin threshold that controls the unlearning strength. The rationale behind (3) is that minimizing it ensures convergence to the negative margin $f_y(\mathbf{x}+\boldsymbol{\delta}) - \max_{k\neq y} f_k(\mathbf{x}+\boldsymbol{\delta}) \to -\tau \leq 0$. Thus, the forget vector $\boldsymbol{\delta}$ enforces unlearning on $\boldsymbol{\theta}$ for $\mathbf{x}$ by making the incorrect prediction ($k \neq y$) have a higher confidence than the original correct prediction y. On the other hand, once the margin becomes negative (indicating that the prediction label has been flipped), the forget objective ℓ_{f} automatically terminates, allowing a balance with the retain objective, which will be introduced later. In our experiments, we find that the forget objective is robust to variations in the *nonnegative* margin parameter τ (see Appendix C). A larger τ value imposes a stricter unlearning requirement by increasing the logit distance from the correct label. For example, we set $\tau = 1$ in our experiments.

Next, we regularize the forget objective (3) with the retain objective, defined as the cross-entropy loss (ℓ_{CE}) over the retain set $\mathcal{D}_{\mathrm{r}}$, along with the ℓ_2 norm of $\boldsymbol{\delta}$ to ensure minimal perturbation required to achieve both the forget and retain objectives. This yields the full unlearning objective in (2):

$$\ell_{\mathrm{MU}}(\boldsymbol{\delta}; \boldsymbol{\theta}_{\mathrm{o}}, \mathcal{D}_{\mathrm{f}}, \mathcal{D}_{\mathrm{r}}) = \ell_{\mathrm{f}}(\boldsymbol{\delta}; \boldsymbol{\theta}_{\mathrm{o}}, \mathcal{D}_{\mathrm{f}}) + \lambda_1 \ell_{\mathrm{CE}}(\boldsymbol{\delta}; \boldsymbol{\theta}_{\mathrm{o}}, \mathcal{D}_{\mathrm{r}}) + \lambda_2 \|\boldsymbol{\delta}\|_2^2, \tag{4}$$

where $\lambda_1 > 0$ and $\lambda_2 > 0$ are the regularization parameters, and $\ell_{\mathrm{CE}}(\boldsymbol{\delta}; \boldsymbol{\theta}_{\mathrm{o}}, \mathcal{D}_{\mathrm{r}})$ denotes the CE loss of the model $\boldsymbol{\theta}_{\mathrm{o}}$ over the perturbed retain set $\{(\mathbf{x}+\boldsymbol{\delta}, y)\}_{(\mathbf{x},y)\in\mathcal{D}_{\mathrm{r}}}$. Integrating (4) into (3), we can then apply stochastic gradient descent (SGD) [61] to optimize the forget vector variable $\boldsymbol{\delta}$.

Compositional Unlearning via Forget Vector Arithmetic. A forget vector defines an unlearning direction in the input space to guide the unlearning process. We explore whether a *new* unlearning direction can be efficiently constructed by interpolating from existing precomputed forget vectors, such as class-wise forget vectors obtained by solving (4) with $\mathcal{D}_{\mathrm{f}}$ defined as each class's training set. This approach is analogous to the concept of task vectors in weight space for model editing [62]. However, to the best of our knowledge, *input-based* task vector arithmetic has not yet been explored in the literature. If forget vectors can be modified and combined using arithmetic operations, such as negation and addition, we can dynamically adjust a model's unlearning behavior without re-solving the optimization problem (4) or any other model-based unlearning problem in (1). We refer to this new unlearning paradigm as *compositional unlearning*, where precomputed class-wise forget vectors can be efficiently combined to generate a new forget vector for each deletion request in the context of random data forgetting.

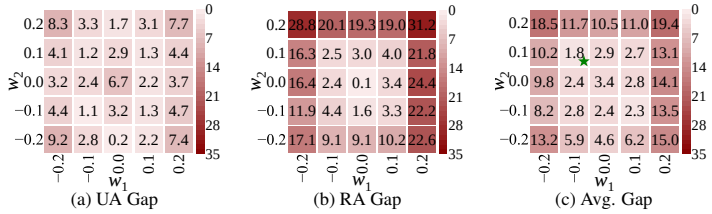
Let $\boldsymbol{\delta}_k$ denote the forget vector used for unlearning data points of class k. Given the set of forget vectors $\{\boldsymbol{\delta}_k\}_{k=1}^K$ for all K classes, we obtain these vectors by solving (4) with $\mathcal{D}_{\mathrm{f}}$ defined as each class's training set, respectively. The forget vector for compositional unlearning is given by

$$\boldsymbol{\delta}(\mathbf{w}) := \sum_{k=1}^{K} (w_k \boldsymbol{\delta}_k), \tag{5}$$

where $\mathbf{w} = [w_1, \ldots w_K]^K$ are the linear combination coefficients to be optimized, which determine the forget vector arithmetic. To determine $\mathbf{w}$, we can minimize (4) with the optimization restricted to the coefficients $\mathbf{w}$. Instead of penalizing the ℓ_2 norm of the forget vector, we penalize the ℓ_2 norm of $\mathbf{w}$ to prevent excessive pixel perturbation. This modifies (4) to the problem $\min_{\mathbf{w}} \ell_{\mathrm{f}}(\boldsymbol{\delta}(\mathbf{w}); \boldsymbol{\theta}_{\mathrm{o}}, \mathcal{D}_{\mathrm{f}}) + \lambda_1 \ell_{\mathrm{CE}}(\boldsymbol{\delta}(\mathbf{w}); \boldsymbol{\theta}_{\mathrm{o}}, \mathcal{D}_{\mathrm{r}}) + \lambda_2 \|\mathbf{w}\|_2^2$. As will be shown later, random data forgetting can be achieved through class-wise forget vector arithmetic (5) by applying the compositional scheme defined by the coefficients $\mathbf{w}$.

To illustrate the effectiveness of forget vector arithmetic, **Fig. 3** shows preliminary results of combining two class-wise forget vectors ($\boldsymbol{\delta}_1$ and $\boldsymbol{\delta}_2$) using a simple scheme $\boldsymbol{\delta}(\mathbf{w}) = w_1\boldsymbol{\delta}_1 + w_2\boldsymbol{\delta}_2$ on (CIFAR-10,

ResNet-18), when forgetting a randomly selected 10% of training points from class "automobile" and class "bird" refering to w_1 and w_2. Rather than optimizing $\mathbf{w}$, we adjust w_1 and w_2 from -0.2 to 0.2 to observe how the performance gap relative to Retrain varies. This evaluation includes the UA (unlearning accuracy) gap on the selected forget data, the RA (retain accuracy) gap on the remaining data, and the average gap across these two metrics. As expected, Fig. 3-(a) and (b) shows a trade-off among these two metrics, where weight configurations that achieve a low UA gap may result in a higher RA gap, and vice versa. Additionally, Fig. 3-(c) shows that moderate weight values of w_1 and w_2 ($-0.1 \leq w_1 \leq 0.1$ and $-0.1 \leq w_2 \leq 0.1$) tent to yield a more balanced average performance, maintaining relatively low gaps across two metrics. A favored weighting scheme can be identified at $w_1 = -0.1$ and $w_2 = 0.1$ as marked by the green star (★), validating the feasibility of arbitrary random data forgetting using our proposed compositional unlearning via the forget vector arithmetic approach.



(a) UA Gap (b) RA Gap (c) Avg. Gap

Figure 3: The performance gap relative to Retrain for class-wise forget vector arithmetic (based on classes "automobile" and "bird") across different combination coefficients w_1 and w_2, when unlearning a randomly selected 10% of training points from these two classes of CIFAR-10. Each cell displays the gap (%) relative to Retrain at a specific weight combination, where a lower value indicates a closer performance to Retrain given a metric. A green star (★) denotes the selected weight combination scheme (w_1 and w_2) that achieves the smallest performance gap relative to Retrain, averaged over both UA Gap and RA Gap.

6 Experiments

6.1 Experiment Setups

Datasets and Models. We focus on MU for image classification, using two datasets: CIFAR-10 [63] and ImageNet-10, a 10-class subset of the original ImageNet [64], for ease of implementation of Retrain (exact unlearning) over ImageNet images as [65, 66]. For these tasks, we use three well-trained image classifiers: ResNet-18 [67] for CIFAR-10, VGG-16 [68] and ViT-[69] for ImageNet.

Unlearning Baselines and Evaluations. In the context of MU for image classification, we consider two scenarios: *class-wise forgetting* and *data-wise forgetting*. In class-wise forgetting, training data from an image class are designated for unlearning, while in random data forgetting, a subset of all-class training points is randomly selected as the forget set, with a specified forget ratio of 10%. To demonstrate the effectiveness of our proposal, we consider 8 MU baseline methods, including ① FT [4], ② RL [2], ③ GA [10], ④ NegGrad+ [55], ⑤ SalUn [5], ⑥ SCRUB [19], ⑦ Class-F [70], and ⑧ SSD [71], where Class-F is only designed for class-wise forgetting.

As described in Sec. 3, unlearning effectiveness is measured using UA (unlearning accuracy) and MIA-Efficacy, while model utility post-unlearning is assessed by RA (retain accuracy) and TA (testing accuracy); For all metrics, being closer to Retrain indicates better performance. It is also worth noting that all existing model-based MU baseline methods ①-⑧ are evaluated on non-perturbed evaluation sets. However, when using our proposed data-based forget vector solution, we need to apply the forget vector to the evaluation sets (including the forget set, retain set, and testing set) in order to assess unlearning effectiveness and utility retention. This evaluation remains fair, as it aligns with the same objective of forgetting targeted data. The key distinction is that the forget vector operates at the input level, whereas model-based MU baselines achieve unlearning by modifying model weights.

To quantify the performance gap with Retrain, we compare each unlearning baseline and our proposal against this exact unlearning method across all metrics. We report an averaged assessment, termed *Averaging (Avg.) Gap*. Unless specified otherwise, all the main experiments (whether class-wise or random data forgetting) are conducted over 10 random trials, with mean performance reported.

Implementation Details of Our Proposal. To solve the forget vector problem (2) with the proposed unlearning objective in (4), we set the retain loss regularization parameter λ_1 as follows: 3 for CIFAR-10, 5 for ImageNet-10 with VGG-16, and 7 for ImageNet-10 with ViT in class-wise forgetting. For random data forgetting, we set λ_1 to 1. The ℓ_2-norm regularization parameter is set to $\lambda_2 = 1$. These hyperparameters are determined through a grid search over the range $[0, 10]$. To optimize (2), we use stochastic gradient descent (SGD) [61] with a momentum factor of 0.9 and an exponential learning

Table 1: Performance overview of various MU methods for image classification under two unlearning scenarios on CIFAR-10 using ResNet-18 and ImageNet-10 using ViT. Since Class-F [70] is specifically designed for class-wise forgetting, its results do not apply to random data forgetting scenarios (n/a). Other results are reported in the format $a_{\pm b}$, where a is the mean and b denotes standard deviation b over 10 independent trials. The performance gap against Retrain is indicated in (•), where a lower value is better. ↑ (or ↓) indicates that a higher (or lower) value is better. The best performance for each metric is highlighted in green, while the second-best performance is highlighted in red.

MU Method	UA↑	MIA-Efficacy↑	RA↑	TA↑	Avg.Gap↓	UA↑	MIA-Efficacy↑	RA↑	TA↑	Avg.Gap↓
	Class-wise Forgetting, CIFAR-10, ResNet-18					Random Data Forgetting, CIFAR-10, ResNet-18				
Retrain	$100.00_{\pm0.00}$(0.00)	$100.00_{\pm0.00}$(0.00)	$99.91_{\pm0.03}$(0.00)	$94.92_{\pm0.15}$(0.00)	**0.00**	$5.50_{\pm0.16}$(0.00)	$11.57_{\pm0.47}$(0.00)	$99.88_{\pm0.05}$(0.00)	$94.24_{\pm0.19}$(0.00)	**0.00**
FT [4]	$5.27_{\pm0.73}$(94.73)	$51.49_{\pm5.07}$(48.51)	$100.0_{\pm0.0}$(0.09)	$95.03_{\pm0.07}$(0.11)	**35.86**	$0.03_{\pm0.03}$(5.47)	$0.75_{\pm0.09}$(10.82)	$99.98_{\pm0.02}$(0.10)	$94.45_{\pm0.14}$(0.21)	**4.15**
RL [2]	$18.87_{\pm7.34}$(81.13)	$98.94_{\pm0.79}$(1.06)	$99.98_{\pm0.0}$(0.07)	$94.51_{\pm0.12}$(0.41)	**20.67**	$0.52_{\pm0.24}$(4.98)	$3.13_{\pm0.55}$(8.44)	$99.87_{\pm0.07}$(0.03)	$93.88_{\pm0.20}$(0.36)	**3.45**
GA [10]	$71.45_{\pm0.35}$(28.55)	$81.7_{\pm0.22}$(18.30)	$98.62_{\pm0.04}$(1.29)	$92.34_{\pm0.02}$(2.58)	**12.68**	$1.56_{\pm3.08}$(3.94)	$2.88_{\pm3.44}$(8.69)	$98.67_{\pm2.74}$(1.21)	$92.84_{\pm2.59}$(1.40)	**3.81**
NegGrad+ [55]	$91.78_{\pm14.66}$(8.22)	$95.81_{\pm7.28}$(4.19)	$98.35_{\pm1.22}$(1.56)	$92.62_{\pm1.34}$(2.30)	**4.07**	$0.97_{\pm1.08}$(4.53)	$2.74_{\pm2.16}$(8.83)	$99.42_{\pm0.87}$(0.46)	$93.38_{\pm1.13}$(0.86)	**3.67**
SalUn [5]	$96.35_{\pm2.14}$(3.65)	$98.64_{\pm0.03}$(1.36)	$98.75_{\pm0.18}$(1.16)	$92.34_{\pm1.54}$(2.58)	**2.19**	$1.73_{\pm0.25}$(3.77)	$6.25_{\pm1.21}$(5.32)	$99.24_{\pm0.09}$(0.64)	$91.03_{\pm0.14}$(3.21)	**3.23**
SCRUB [19]	$93.45_{\pm2.33}$(6.55)	$96.38_{\pm1.72}$(3.62)	$99.95_{\pm0.0}$(0.04)	$94.56_{\pm0.07}$(0.36)	**2.64**	$0.61_{\pm0.31}$(4.89)	$3.69_{\pm0.54}$(7.88)	$99.76_{\pm0.18}$(0.12)	$93.91_{\pm0.19}$(0.33)	**3.31**
Class-F [70]	$90.18_{\pm1.20}$(9.82)	$86.15_{\pm2.67}$(13.85)	$91.25_{\pm0.05}$(8.66)	$85.45_{\pm0.19}$(9.47)	**10.45**	n/a	n/a	n/a	n/a	n/a
SSD [71]	$96.05_{\pm0.45}$(3.95)	$98.00_{\pm0.00}$(2.00)	$97.77_{\pm0.20}$(2.14)	$92.23_{\pm0.58}$(2.69)	**2.70**	$5.54_{\pm0.00}$(0.04)	$7.80_{\pm0.00}$(3.77)	$94.86_{\pm0.00}$(5.02)	$88.28_{\pm0.00}$(5.96)	**3.70**
Ours	$97.88_{\pm0.27}$(2.12)	$99.60_{\pm0.15}$(0.40)	$97.25_{\pm0.24}$(2.66)	$90.90_{\pm0.32}$(4.02)	**2.30**	$2.61_{\pm0.49}$(2.89)	$8.20_{\pm1.17}$(3.00)	$97.33_{\pm0.47}$(2.55)	$90.97_{\pm0.38}$(3.27)	**2.92**
	Class-wise Forgetting, ImageNet-10, ViT					Random Data Forgetting, ImageNet-10, ViT				
Retrain	$100_{\pm0.00}$(0.00)	$100.00_{\pm0.00}$(0.00)	$99.97_{\pm0.03}$(0.00)	$99.85_{\pm0.01}$(0.00)	**0.00**	$1.41_{\pm0.06}$(0.00)	$93.57_{\pm0.00}$(0.00)	$99.07_{\pm0.01}$(0.00)	$99.27_{\pm0.01}$(0.00)	**0.00**
FT [4]	$42.79_{\pm7.51}$(57.21)	$40.78_{\pm10.68}$(59.22)	$99.96_{\pm0.01}$(0.01)	$99.61_{\pm0.10}$(0.24)	**29.17**	$1.38_{\pm0.16}$(0.03)	$96.40_{\pm0.31}$(2.83)	$99.60_{\pm0.09}$(0.53)	$99.10_{\pm0.30}$(0.17)	**0.89**
RL [2]	$88.15_{\pm1.62}$(11.85)	$96.50_{\pm1.56}$(3.50)	$99.93_{\pm0.02}$(0.04)	$99.89_{\pm0.11}$(0.04)	**3.84**	$2.62_{\pm0.80}$(1.21)	$93.75_{\pm0.52}$(0.18)	$98.38_{\pm0.18}$(0.69)	$95.05_{\pm0.26}$(4.20)	**1.57**
GA [10]	$28.05_{\pm7.05}$(71.95)	$58.73_{\pm5.83}$(41.27)	$99.97_{\pm0.01}$(0.00)	$99.70_{\pm0.10}$(0.15)	**28.34**	$0.82_{\pm0.16}$(0.59)	$96.77_{\pm1.10}$(3.20)	$99.60_{\pm0.09}$(0.53)	$99.53_{\pm0.09}$(0.26)	**1.15**
NegGrad+ [55]	$10.90_{\pm2.87}$(89.10)	$79.30_{\pm4.58}$(20.70)	$99.98_{\pm0.00}$(0.01)	$99.78_{\pm0.00}$(0.07)	**27.46**	$1.97_{\pm0.72}$(0.56)	$95.48_{\pm1.67}$(1.91)	$98.09_{\pm0.94}$(0.98)	$98.08_{\pm0.75}$(1.19)	**1.16**
SalUn [5]	$93.27_{\pm1.50}$(6.73)	$94.00_{\pm1.06}$(6.00)	$98.22_{\pm0.75}$(1.75)	$98.00_{\pm0.06}$(1.85)	**4.08**	$0.67_{\pm0.19}$(0.74)	$95.80_{\pm0.0}$(2.23)	$99.65_{\pm0.07}$(0.58)	$98.27_{\pm0.09}$(0.00)	**1.14**
SCRUB [19]	$99.10_{\pm0.50}$(0.90)	$95.67_{\pm1.77}$(4.33)	$98.90_{\pm0.10}$(1.07)	$99.33_{\pm0.31}$(0.52)	**1.71**	$0.85_{\pm0.24}$(0.56)	$95.90_{\pm0.95}$(2.33)	$99.58_{\pm0.23}$(0.51)	$99.20_{\pm0.11}$(0.07)	**0.87**
Class-F [70]	$28.62_{\pm7.85}$(71.38)	$55.10_{\pm2.30}$(44.90)	$77.50_{\pm1.62}$(22.47)	$75.22_{\pm0.56}$(24.63)	**40.85**	n/a	n/a	n/a	n/a	n/a
SSD [71]	$90.35_{\pm1.65}$(9.65)	$60.15_{\pm1.85}$(39.85)	$98.43_{\pm0.55}$(1.54)	$98.33_{\pm0.56}$(1.52)	**13.14**	$1.12_{\pm0.04}$(0.29)	$94.15_{\pm0.35}$(0.58)	$98.97_{\pm0.00}$(0.10)	$99.40_{\pm0.00}$(0.13)	**0.28**
Ours	$95.92_{\pm0.27}$(4.08)	$99.40_{\pm0.14}$(0.60)	$99.13_{\pm0.04}$(0.84)	$99.26_{\pm0.28}$(0.59)	**1.53**	$1.08_{\pm0.39}$(0.33)	$91.40_{\pm1.30}$(2.17)	$98.97_{\pm0.35}$(0.10)	$99.10_{\pm0.50}$(0.17)	**0.69**

rate scheduler, decaying at a rate of 0.9 per iteration. Additionally, the batch size is set to 256, with a maximum of 40 optimization iterations for both two datasets. To solve the compositional unlearning problem (5), we use a similar setup, setting both λ_1 and λ_2 to 1.

6.2 Experiment Results

Overview Performance of Forget Vector. In Tab. 1, we compare the performance of our forget vector approach with other model-based MU methods across the metrics: UA, RA, TA, MIA-Efficacy, and Avg. Gap vs. Retrain. We highlight two key observations below. **First**, in terms of unlearning effectiveness (UA and MIA-Efficacy), the data perturbation-based forget vector demonstrates highly competitive performance compared to model update-based MU baselines, mostly ranking among the top two methods with the smallest performance gap relative to Retrain (as evidenced by Avg. Gap). The advantage of the forget vector is particularly evident in MIA-Efficacy, where it usually achieves the closest results to Retrain. **Second**, in terms of model utility post-unlearning (RA and TA), the forget vector generally leads to a larger performance drop than other methods. This is not surprising, as the forget vector is achieved through data perturbations. However, considering the gain in unlearning effectiveness, the Avg. Gap with Retrain shows that the forget vector remains competitive, ranking among the top two unlearning methods. **Third**, unlearning methods (including Retrain) do not exhibit the same level of distinctiveness in random data forgetting as it does in class-wise forgetting. This is because in random data forgetting, the retain data could have sufficiently represented the distribution of the forget data, making it more challenging for MIA to distinguish forgotten samples from retained ones. Besides, the corresponding results of various MU methods on ImageNet-10 using VGG-16 can be found in Appendix D.

Transferability of Forget Vector to Unseen Forget Data. Conventionally, unlearning effectiveness is typically measured on the original forget set ($\mathcal{D}_\mathrm{f}$). However, with the data perturbation-based forget vector, it is also interesting to investigate its unlearning transferability when applied to a new, previously unseen forget set (denoted as $\mathcal{D}_\mathrm{f}{}'$) that share similarities with $\mathcal{D}_\mathrm{f}$ and are equally appropriate for unlearning. In the context of class-wise forgetting, we consider $\mathcal{D}'_\mathrm{f}$ using the testing data from the class targeted for unlearning, where unlearning performance should align closely with Retrain since the test-time data to forget share the same distribution with the training set. In the context of data forgetting, we obtain $\mathcal{D}'_\mathrm{f}$ by applying the data corruption operation GN1 and ET1 used in Fig. 2 to perturb $\mathcal{D}_\mathrm{f}$ (last two columns of Tab. 2), where Retrain is no longer the gold standard as training data distribution excludes these shifts, allowing higher UA for better unlearning. As observed

Table 2: UA (%) of forget vector when transferred to unseen forget sets curated under 3 scenarios on (CIFAR-10, ResNet-18). The results are presented in the same format as Table 1.

MU Method	D'_f (Class-wise) from testing set	D'_f (Random Data) perturbed by GN1	D'_f (Random Data) perturbed by ET1
Retrain	$100.00_{\pm0.00}$(0.00)	$64.73_{\pm3.36}$(0.00)	$81.43_{\pm0.28}$(0.00)
FT [4]	$21.44_{\pm1.11}$(78.56)	$56.75_{\pm1.40}$(7.98)	$81.79_{\pm0.18}$(0.36)
RL [2]	$27.90_{\pm5.70}$(72.10)	$61.84_{\pm2.45}$(2.89)	$81.15_{\pm0.20}$(0.28)
GA [10]	$73.95_{\pm0.60}$(26.05)	$55.20_{\pm2.59}$(9.53)	$82.17_{\pm0.76}$(1.28)
NegGrad+ [55]	$93.86_{\pm10.93}$(6.14)	$57.81_{\pm1.76}$(6.92)	$81.73_{\pm0.65}$(0.30)
SalUn [5]	$97.55_{\pm1.37}$(2.45)	$73.15_{\pm4.25}$(8.42)	$80.27_{\pm3.56}$(1.16)
SCRUB [19]	$93.65_{\pm2.65}$(6.35)	$61.95_{\pm0.86}$(1.85)	$81.18_{\pm0.62}$(0.25)
Class-F [70]	$89.14_{\pm5.13}$(10.66)	n/a	n/a
SSD [71]	$98.70_{\pm0.03}$(1.30)	$80.34_{\pm0.01}$(15.91)	$84.70_{\pm0.02}$(3.27)
Ours	$98.26_{\pm0.35}$(1.74)	$78.32_{\pm1.03}$(13.59)	$85.03_{\pm0.91}$(3.60)

in Tab. 2, the unlearning performance of the forget vector remains effective when evaluated on $\mathcal{D}_f'$. Among the model update-based unlearning baselines, current SOTA methods such as SCRUB [19], SalUn [5] and SSD [71] also demonstrate generalization to $\mathcal{D}_f'$, compared to simpler MU methods like FT, RL, and GA, which show lower transferability.

Compositional Unlearning by Class-wise Forget Vectors. Next, we demonstrate the effectiveness of compositional unlearning via forget vector arithmetic (termed CU-FV). Given pre-computed class-wise forget vectors, we apply their linear combination as defined in (5) to achieve random data forgetting. Tab. 3 compares the performance of CU-FV with Retrain (the exact unlearning method) and the direct forget vector approach (FV) applied to the targeted forget set. Interestingly, we observe that CU-FV achieves the overall performance comparable to FV, as indicated by similar Avg. Gap values. Unlike FV, CU-FV optimizes only the class-wise coefficients in (5), resulting in a much smaller optimization space than FV. However, from UA and MIA-Efficacy metrics, we find that unlearning effectiveness is easier to maintain since unlearning typically targets a smaller subset of data points. In contrast, model utility (RA and TA) may decline more with CU-FV than with FV.

Table 3: Compositional unlearning on CIFAR-10 and ImageNet-10 for random data forgetting, where FV represents the original setting of forget vector that is directly learned based on a targeted forget set, and CU-FV denotes compositional unlearning achieved via pre-learned class-wise forget vector arithmetic. The results are presented in the same format as Table 1.

Module	MU Method	UA↑	MIA-Efficacy↑	RA↑	TA↑	Avg.Gap↓
CIFAR-10 ResNet-18	Retrain	$5.50_{\pm 0.16}$(0.00)	$11.57_{\pm 0.47}$(0.00)	$94.24_{\pm 0.19}$(0.00)	$99.88_{\pm 0.05}$(0.00)	**0.00**
	FV	$2.61_{\pm 0.49}$(2.89)	$8.26_{\pm 1.17}$(3.00)	$90.97_{\pm 0.38}$(3.27)	$97.33_{\pm 0.47}$(2.55)	**2.92**
	CU-FV	$5.36_{\pm 0.60}$(0.14)	$9.76_{\pm 0.91}$(1.81)	$88.60_{\pm 0.59}$(5.64)	$94.93_{\pm 0.64}$(4.95)	**3.16**
ImageNet-10 VGG-16	Retrain	$4.05_{\pm 0.45}$(0.00)	$6.60_{\pm 1.07}$(0.00)	$96.33_{\pm 0.38}$(0.00)	$99.48_{\pm 0.07}$(0.00)	**0.00**
	FV	$2.27_{\pm 0.50}$(1.78)	$6.13_{\pm 1.40}$(0.47)	$95.82_{\pm 0.48}$(0.51)	$98.29_{\pm 0.32}$(1.19)	**0.99**
	CU-FV	$2.27_{\pm 1.18}$(1.78)	$4.95_{\pm 1.86}$(1.65)	$91.41_{\pm 1.25}$(4.92)	$97.93_{\pm 1.09}$(1.55)	**2.47**
ImageNet-10 ViT	Retrain	$1.41_{\pm 0.06}$(0.00)	$99.07_{\pm 0.01}$(0.00)	$93.57_{\pm 0.00}$(0.00)	$99.27_{\pm 0.01}$(0.00)	**0.00**
	FV	$1.08_{\pm 0.39}$(0.33)	$98.97_{\pm 0.35}$(0.10)	$91.40_{\pm 1.30}$(2.17)	$99.10_{\pm 0.50}$(0.17)	**0.69**
	CU-FV	$1.62_{\pm 0.05}$(0.21)	$98.55_{\pm 0.14}$(0.52)	$92.09_{\pm 0.25}$(1.48)	$98.50_{\pm 0.01}$(0.77)	**0.75**

Assessing Forget Vector via An Input Saliency Lens. In Fig. 4, we explore the impact of the forget vector on unlearning and utility retention through input saliency map, using Grad-CAM (Gradient-weighted Class Activation Mapping) [72]. Using Grad-CAM, we visualize the salient pixels (*i.e.*, regions most influential to model prediction) for forget images under different unlearning scenarios: (1) the original model (without unlearning), (2) Retrain, (3) SCRUB-based unlearning, and (4) forget vector-based unlearning. In the first three scenarios, we obtain the input saliency maps on raw forget images without the addition of the forget vector, while in the last scenario, the input saliency is applied to images perturbed by the forget vector. As seen in fig. 4 the forget vector and Retrain effectively unlearn the target input images, evident from the significant shifts in saliency regions compared to those in the original model. In contrast, the MU baseline SCRUB shows minimal saliency shifts, failing to adequately forget the target data in the last row. More visualization results on both forget and retain images can be found in Appendix G.

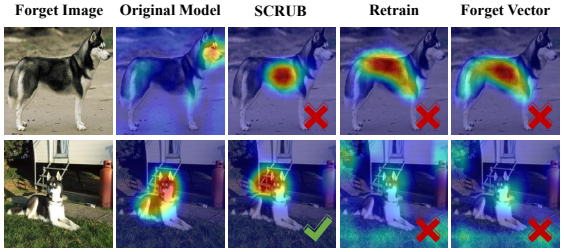


Figure 4: Gradient-based saliency map visualized via Grad-CAM for different MU methods against **forget images**. The highlighted areas (marked in red) indicate regions most influential to model prediction, and the red cross mark (✖) indicates that corresponding methods effectively unlearn the input forget images, while the check (✔) signifies the opposite.

Ablation Studies. In Appendix E, we provide additional ablation studies on the sensitivity of the unlearning-retaining regularization parameter λ_1 in (4). Moreover, we perform the efficiency analysis of forget vector calculation with those of other model-based unlearning methods in Appendix F.

7 Conclusion and Limitations

In this paper, we introduce a novel, data-based approach to machine unlearning (MU) in image classification, termed the *forget vector*. Unlike traditional model-based MU methods that require retraining or fine-tuning model weights, our approach shows that input-agnostic data perturbations can effectively achieve unlearning objectives. Our method demonstrates competitive performance relative to model-based approximate unlearning techniques. Furthermore, we showcase the potential of compositional unlearning: new forget vectors for unseen tasks, such as unlearning arbitrary subsets across classes, can be generated through simple arithmetic operations, like linear combinations of class-specific forget vectors. Extensive experiments confirm the effectiveness and adaptability of our optimized forget vector. Moreover, broader impacts and limitations discussion can be found in Appendix H.

References

[1] Chris Jay Hoofnagle, Bart Van Der Sloot, and Frederik Zuiderveen Borgesius. The european union general data protection regulation: what it is and what it means. *Information & Communications Technology Law*, 28(1):65–98, 2019.

[2] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9304–9312, 2020.

[3] Samuele Poppi, Sara Sarto, Marcella Cornia, Lorenzo Baraldi, and Rita Cucchiara. Multi-class explainable unlearning for image classification via weight filtering. *arXiv preprint arXiv:2304.02049*, 2023.

[4] Alexander Warnecke, Lukas Pirch, Christian Wressnegger, and Konrad Rieck. Machine unlearning of features and labels. *arXiv preprint arXiv:2108.11577*, 2021.

[5] Chongyu Fan, Jiancheng Liu, Yihua Zhang, Dennis Wei, Eric Wong, and Sijia Liu. Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation. *arXiv preprint arXiv:2310.12508*, 2023.

[6] Yushun Dong, Binchi Zhang, Zhenyu Lei, Na Zou, and Jundong Li. Idea: A flexible framework of certified unlearning for graph neural networks. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 621–630, 2024.

[7] Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified data removal from machine learning models. *arXiv preprint arXiv:1911.03030*, 2019.

[8] Anvith Thudi, Hengrui Jia, Ilia Shumailov, and Nicolas Papernot. On the necessity of auditable algorithmic definitions for machine unlearning. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 4007–4022, 2022.

[9] Laura Graves, Vineel Nagisetty, and Vijay Ganesh. Amnesiac machine learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11516–11524, 2021.

[10] 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)*, pages 303–319. IEEE, 2022.

[11] Alexander Becker and Thomas Liebig. Evaluating machine unlearning via epistemic uncertainty. *arXiv preprint arXiv:2208.10836*, 2022.

[12] Zachary Izzo, Mary Anne Smart, Kamalika Chaudhuri, and James Zou. Approximate data deletion from machine learning models. In *International Conference on Artificial Intelligence and Statistics*, pages 2008–2016. PMLR, 2021.

[13] Sungmin Cha, Sungjun Cho, Dasol Hwang, Honglak Lee, Taesup Moon, and Moontae Lee. Learning to unlearn: Instance-wise unlearning for pre-trained classifiers. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 11186–11194, 2024.

[14] Min Chen, Weizhuo Gao, Gaoyang Liu, Kai Peng, and Chen Wang. Boundary unlearning: Rapid forgetting of deep networks via shifting the decision boundary. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7766–7775, 2023.

[15] 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.

[16] Vasileios Perifanis, Efstathios Karypidis, Nikos Komodakis, and Pavlos Efraimidis. Sftc: Machine unlearning via selective fine-tuning and targeted confusion. In *European Interdisciplinary Cybersecurity Conference*, pages 29–36, 2024.

[17] Kongyang Chen, Zixin Wang, Bing Mi, Waixi Liu, Shaowei Wang, Xiaojun Ren, and Jiaxing Shen. Machine unlearning in large language models. *arXiv preprint arXiv:2404.16841*, 2024.

[18] Aditya Golatkar, Alessandro Achille, Avinash Ravichandran, Marzia Polito, and Stefano Soatto. Mixed-privacy forgetting in deep networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 792–801, 2021.

[19] Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. Towards unbounded machine unlearning. *Advances in neural information processing systems*, 36, 2024.

[20] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019.

[21] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[22] Hyojin Bahng, Ali Jahanian, Swami Sankaranarayanan, and Phillip Isola. Visual prompting: Modifying pixel space to adapt pre-trained models. *arXiv preprint arXiv:2203.17274*, 3(11-12):3, 2022.

[23] Aochuan Chen, Yuguang Yao, Pin-Yu Chen, Yihua Zhang, and Sijia Liu. Understanding and improving visual prompting: A label-mapping perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19133–19143, 2023.

[24] Changdae Oh, Hyeji Hwang, Hee-young Lee, YongTaek Lim, Geunyoung Jung, Jiyoung Jung, Hosik Choi, and Kyungwoo Song. Blackvip: Black-box visual prompting for robust transfer learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24224–24235, 2023.

[25] Gamaleldin F Elsayed, Ian Goodfellow, and Jascha Sohl-Dickstein. Adversarial reprogramming of neural networks. *arXiv preprint arXiv:1806.11146*, 2018.

[26] Guanhua Zhang, Yihua Zhang, Yang Zhang, Wenqi Fan, Qing Li, Sijia Liu, and Shiyu Chang. Fairness reprogramming. *Advances in Neural Information Processing Systems*, 35:34347–34362, 2022.

[27] Pin-Yu Chen. Model reprogramming: Resource-efficient cross-domain machine learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 22584–22591, 2024.

[28] Weiqi Wang, Zhiyi Tian, Chenhan Zhang, and Shui Yu. Machine unlearning: A comprehensive survey. *arXiv preprint arXiv:2405.07406*, 2024.

[29] Chao Pan, Jin Sima, Saurav Prakash, Vishal Rana, and Olgica Milenkovic. Machine unlearning of federated clusters. *arXiv preprint arXiv:2210.16424*, 2022.

[30] Guihong Li, Hsiang Hsu, Chun-Fu Chen, and Radu Marculescu. Machine unlearning for image-to-image generative models. *arXiv preprint arXiv:2402.00351*, 2024.

[31] Jack Foster, Stefan Schoepf, and Alexandra Brintrup. Loss-free machine unlearning. *arXiv preprint arXiv:2402.19308*, 2024.

[32] 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*, pages 20147–20155, 2023.

[33] Varun Gupta, Christopher Jung, Seth Neel, Aaron Roth, Saeed Sharifi-Malvajerdi, and Chris Waites. Adaptive machine unlearning. *Advances in Neural Information Processing Systems*, 34:16319–16330, 2021.

[34] Jimmy Z Di, Jack Douglas, Jayadev Acharya, Gautam Kamath, and Ayush Sekhari. Hidden poison: Machine unlearning enables camouflaged poisoning attacks. In *NeurIPS ML Safety Workshop*, 2022.

[35] Ruizhe Chen, Jianfei Yang, Huimin Xiong, Jianhong Bai, Tianxiang Hu, Jin Hao, Yang Feng, Joey Tianyi Zhou, Jian Wu, and Zuozhu Liu. Fast model debias with machine unlearning. *Advances in Neural Information Processing Systems*, 36, 2024.

[36] Yimeng Zhang, Xin Chen, Jinghan Jia, Yihua Zhang, Chongyu Fan, Jiancheng Liu, Mingyi Hong, Ke Ding, and Sijia Liu. Defensive unlearning with adversarial training for robust concept erasure in diffusion models. *arXiv preprint arXiv:2405.15234*, 2024.

[37] 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*, 36, 2024.

[38] Nicholas Pochinkov and Nandi Schoots. Dissecting language models: Machine unlearning via selective pruning. *arXiv preprint arXiv:2403.01267*, 2024.

[39] Weijia Shi, Jaechan Lee, Yangsibo Huang, Sadhika Malladi, Jieyu Zhao, Ari Holtzman, Daogao Liu, Luke Zettlemoyer, Noah A Smith, and Chiyuan Zhang. Muse: Machine unlearning six-way evaluation for language models. *arXiv preprint arXiv:2407.06460*, 2024.

[40] Lingzhi Wang, Xingshan Zeng, Jinsong Guo, Kam-Fai Wong, and Georg Gottlob. Selective forgetting: Advancing machine unlearning techniques and evaluation in language models. *arXiv preprint arXiv:2402.05813*, 2024.

[41] Lingzhi Wang, Tong Chen, Wei Yuan, Xingshan Zeng, Kam-Fai Wong, and Hongzhi Yin. Kga: A general machine unlearning framework based on knowledge gap alignment. *arXiv preprint arXiv:2305.06535*, 2023.

[42] Sijia Liu, Yuanshun Yao, Jinghan Jia, Stephen Casper, Nathalie Baracaldo, Peter Hase, Xiaojun Xu, Yuguang Yao, Hang Li, Kush R Varshney, et al. Rethinking machine unlearning for large language models. *arXiv preprint arXiv:2402.08787*, 2024.

[43] Xunkai Li, Yulin Zhao, Zhengyu Wu, Wentao Zhang, Rong-Hua Li, and Guoren Wang. Towards effective and general graph unlearning via mutual evolution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 13682–13690, 2024.

[44] Min Du, Zhi Chen, Chang Liu, Rajvardhan Oak, and Dawn Song. Lifelong anomaly detection through unlearning. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, pages 1283–1297, 2019.

[45] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision*, pages 709–727. Springer, 2022.

[46] Wenhao Wang, Yifan Sun, Wei Li, and Yi Yang. Transhp: Image classification with hierarchical prompting. *Advances in Neural Information Processing Systems*, 36:28187–28200, 2023.

[47] Weihuang Liu, Xi Shen, Chi-Man Pun, and Xiaodong Cun. Explicit visual prompting for low-level structure segmentations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19434–19445, 2023.

[48] Mir Rayat Imtiaz Hossain, Mennatullah Siam, Leonid Sigal, and James J Little. Visual prompting for generalized few-shot segmentation: A multi-scale approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23470–23480, 2024.

[49] Yichi Zhang, Yinpeng Dong, Siyuan Zhang, Tianzan Min, Hang Su, and Jun Zhu. Exploring the transferability of visual prompting for multimodal large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26562–26572, 2024.

[50] Yun-Yun Tsai, Pin-Yu Chen, and Tsung-Yi Ho. Transfer learning without knowing: Reprogramming black-box machine learning models with scarce data and limited resources. In *International Conference on Machine Learning*, pages 9614–9624. PMLR, 2020.

[51] Chao-Han Huck Yang, Yun-Yun Tsai, and Pin-Yu Chen. Voice2series: Reprogramming acoustic models for time series classification. In *International conference on machine learning*, pages 11808–11819. PMLR, 2021.

[52] Igor Melnyk, Vijil Chenthamarakshan, Pin-Yu Chen, Payel Das, Amit Dhurandhar, Inkit Padhi, and Devleena Das. Reprogramming pretrained language models for antibody sequence infilling. In *International Conference on Machine Learning*, pages 24398–24419. PMLR, 2023.

[53] Jindong Wang, Wenjie Feng, Yiqiang Chen, Han Yu, Meiyu Huang, and Philip S Yu. Visual domain adaptation with manifold embedded distribution alignment. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 402–410, 2018.

[54] Jafar Tahmoresnezhad and Sattar Hashemi. Visual domain adaptation via transfer feature learning. *Knowledge and information systems*, 50:585–605, 2017.

[55] Meghdad Kurmanji, Eleni Triantafillou, and Peter Triantafillou. Machine unlearning in learned databases: An experimental analysis. *Proceedings of the ACM on Management of Data*, 2(1):1–26, 2024.

[56] Jinghan Jia, Jiancheng Liu, Parikshit Ram, Yuguang Yao, Gaowen Liu, Yang Liu, Pranay Sharma, and Sijia Liu. Model sparsity can simplify machine unlearning. *Advances in neural information processing systems*, 36, 2023.

[57] Hyojin Bahng, Ali Jahanian, Swami Sankaranarayanan, and Phillip Isola. Exploring visual prompts for adapting large-scale models. *arXiv preprint arXiv:2203.17274*, 2022.

[58] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8340–8349, 2021.

[59] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[60] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. Ieee, 2017.

[61] Shun-ichi Amari. Backpropagation and stochastic gradient descent method. *Neurocomputing*, 5(4-5):185–196, 1993.

[62] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022.

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

[64] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[65] Zhizhong Huang, Jie Chen, Junping Zhang, and Hongming Shan. Learning representation for clustering via prototype scattering and positive sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(6):7509–7524, 2022.

[66] Yaling Tao, Kentaro Takagi, and Kouta Nakata. Clustering-friendly representation learning via instance discrimination and feature decorrelation. *arXiv preprint arXiv:2106.00131*, 2021.

[67] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

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

[69] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[70] Sangamesh Kodge, Gobinda Saha, and Kaushik Roy. Deep unlearning: Fast and efficient gradient-free class forgetting. *Transactions on Machine Learning Research*, 2024.

[71] Jack Foster, Stefan Schoepf, and Alexandra Brintrup. Fast machine unlearning without retraining through selective synaptic dampening. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 12043–12051, 2024.

[72] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[73] Liwei Song and Prateek Mittal. Systematic evaluation of privacy risks of machine learning models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2615–2632, 2021.

[74] Fernando Pérez-Cruz. Kullback-leibler divergence estimation of continuous distributions. In *2008 IEEE international symposium on information theory*, pages 1666–1670. IEEE, 2008.

[75] Yihua Zhang, Hongkang Li, Yuguang Yao, Aochuan Chen, Shuai Zhang, Pin-Yu Chen, Meng Wang, and Sijia Liu. Visual prompting reimagined: The power of activation prompts. 2024.

Appendix

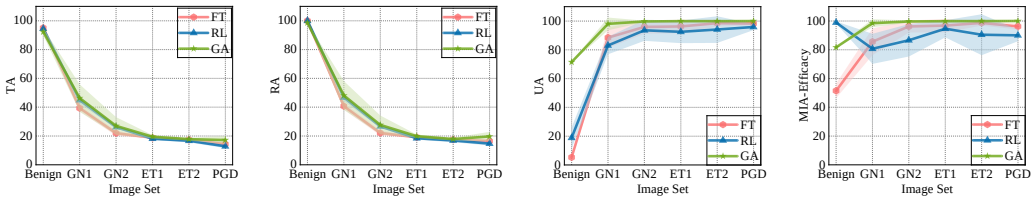
A Details of MIA Implementation

To evaluate the effectiveness of the unlearning process, MIA is implemented following [56] using a prediction confidence-based attack method [73], which comprises a training phase and a testing phase in its computation. Specifically, a balanced dataset is first formed by sampling data points from retain set $\mathcal{D}_\mathrm{r}$ and test set $\mathcal{D}_\mathrm{t}$, ensuring the distinction from the forget set $\mathcal{D}_\mathrm{f}$. Then, a MIA predictor is trained utilizing such balanced dataset. Thereafter, MIA-Efficacy can be calculated by applying the trained MIA predictor to the unlearned model $\boldsymbol{\theta}_\mathrm{u}$ on the forget set $\mathcal{D}_\mathrm{f}$. Essentially, the goal is to determine how many samples in $\mathcal{D}_\mathrm{f}$ can be accurately identified as non-training data with respect to $\boldsymbol{\theta}_\mathrm{u}$ by the MIA model. Formally, MIA-Efficacy is defined as follows,

$$\text{MIA-Efficacy} = N_{tn}/|\mathcal{D}_\mathrm{f}|, \tag{6}$$

where N_{tn} represents the total number of true negative samples in the forget set $\mathcal{D}_\mathrm{f}$ predicted by the trained MIA model, *i.e.*, the number of forgetting samples classified as non-training examples.

B Generalization of MU to Forget Data Shifts



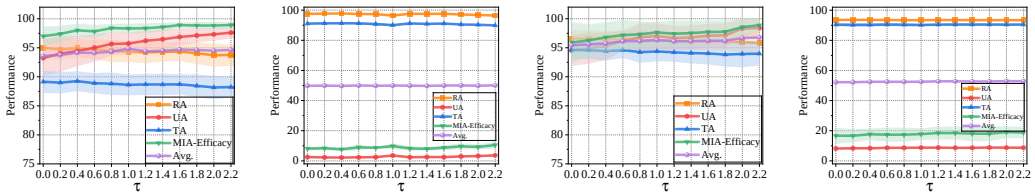
(a) TA (b) RA (c) UA (d) MIA-Efficacy

Figure A1: The performance of class-wise forgetting on (ResNet-18, CIFAR-10) using the unlearning method FT, RL and GA, evaluated on both benign evaluation sets (Benign) and perturbed sets, which include (1) Gaussian noise (GN) with a standard deviation of 0.08 (termed GN1), (2) GN with a standard deviation of 0.2 (termed GN2), (3) Elastic transformation (ET) with parameters (488, 170.8, 24.4) regarding intensity, smoothing, and offset (termed ET1), (4) ET with parameters (488, 19.52, 48.8) (termed ET2), and (5) adversarial perturbations from a 7-step PGD attack with strength $\epsilon = 8/255$. The unlearning performance metrics are reported as (a) TA (testing accuracy), (b) RA (retain accuracy), (c) UA (unlearning accuracy), and (d) MIA-Efficacy, as defined in Sec.3 of main paper. The average performance is reported over 10 independent trials, where each trial focuses on forgetting one specific class from CIFAR-10. Shaded regions indicate the performance variance.

In **Fig. A1**, we provide additional evaluations of other approximate unlearning methods, including FT, RL, and GA against Gaussian noise at test time with standard deviations of 0.08 and 0.2 [20], and two types of Elastic transformations with parameters $(488, 170.8, 24.4)$ and $(488, 19.52, 48.8)$ regarding intensity, smoothing and offset for moderate and high-intensity distortions [20], as well as a 7-step PGD attack with perturbation strength $\epsilon = 8/255$ [21]. The experiments are conducted on the CIFAR-10 dataset using ResNet-18 for the image classification task. As can be seen, the experimental results presented in Fig. A1 are consistent with the findings in Sec.4 of the main paper, further reinforcing the validity of our conclusions. Specifically, as shown in Fig. A1-(a) and (b), model utility, measured by RA (retain accuracy) and TA (testing accuracy), decreases when external perturbations are applied to the evaluation sets compared to its original performance without perturbations. Meanwhile, Fig. A1-(c) and (d) show that unlearning effectiveness measured by UA (unlearning accuracy) and MIA-Efficacy, remains stable despite the presence of these perturbations on the forget set.

C Parameter Sensitivity Analysis: Prediction Margin τ in Forget Vector Loss

In **Fig. A2**, we provide the sensitivity analysis of τ on two datasets regarding two forgetting scenarios: class-wise forgetting and random data forgetting, where we varied τ from 0.0 to 2.2 with a step of 0.2. As can be seen, the forget objective is robust to variations in the nonnegative margin parameter τ. When τ is set to 1, the performance across four metrics achieves an optimal tradeoff. Therefore, we choose $\tau = 1$ for our experiments.



(a) CIFAR-10 Class-wise Forgetting (b) CIFAR-10 Random Data Forgetting (b) ImageNet-10 Class-wise Forgetting (d) ImageNet-10 Random Data Forgetting

Figure A2: Sensitivity analysis of the nonnegative margin parameter τ for image classification under two unlearning scenarios on CIFAR-10 and ImageNet-10 using ResNet-18 and VGG-16, respectively. The unlearning performance metrics are reported as (a) TA (testing accuracy, blue curve), (b) RA (retain accuracy, orange curve), (c) UA (unlearning accuracy, red curve), and (d) MIA-Efficacy (green curve), (e) the average (Avg.) performance across TA, RA, UA, and MIA-Efficacy (purple curve). For class-wise forgetting scenario, the performance is averaged over 10 independent trials, with each trial focusing on forgetting one specific class from the dataset. Similarly, for random-data forgetting scenario, the performance is reported across 10 independent trials, where each trial targets the forgetting of a random subset of the dataset. The shaded regions represent the variance in performance across trials.

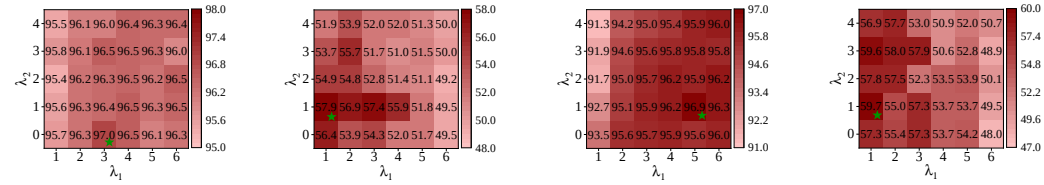
D Performance on ImageNet-10 using VGG-16

Table A1: Performance overview of various MU methods for image classification under two unlearning scenarios on ImageNet-10 using VGG-16. Since Class-F [70] is specifically designed for class-wise forgetting, its results do not apply to random data forgetting scenarios (n/a). Other results are reported in the format $a_{\pm b}$, where a is the mean and b denotes standard deviation b over 10 independent trials. The performance gap against Retrain is indicated in (•), where a lower value is better. ↑ (or ↓) indicates that a higher (or lower) value is better. The best performance for each metric is highlighted in green, while the second-best performance is highlighted in red.

MU Method	UA↑	MIA-Efficacy↑	RA↑	TA↑	**Avg.Gap↓**	UA↑	MIA-Efficacy↑	RA↑	TA↑	**Avg.Gap↓**
	Class-wise Forgetting, ImageNet-10, VGG-16					Random Data Forgetting, ImageNet-10, VGG-16				
Retrain	$100.00_{\pm 0.00}$(0.00)	$100.00_{\pm 0.00}$(0.00)	$99.66_{\pm 0.16}$(0.00)	$97.11_{\pm 0.82}$(0.00)	**0.00**	$4.05_{\pm 0.45}$(0.00)	$6.60_{\pm 1.07}$(0.00)	$99.48_{\pm 0.07}$(0.00)	$96.33_{\pm 0.38}$(0.00)	**0.00**
FT [4]	$39.66_{\pm 4.73}$(60.34)	$55.76_{\pm 7.26}$(44.24)	$99.78_{\pm 0.03}$(0.13)	$97.27_{\pm 0.35}$(2.35)	**26.77**	$1.35_{\pm 0.32}$(2.70)	$4.67_{\pm 1.61}$(1.93)	$99.36_{\pm 0.28}$(0.12)	$96.54_{\pm 0.59}$(0.21)	**1.24**
RL [2]	$76.58_{\pm 11.64}$(23.42)	$46.04_{\pm 33.71}$(53.96)	$99.28_{\pm 0.20}$(0.63)	$96.91_{\pm 0.55}$(1.99)	**20.00**	$2.96_{\pm 0.42}$(1.09)	$12.85_{\pm 4.25}$(6.25)	$99.19_{\pm 0.16}$(0.29)	$95.50_{\pm 0.90}$(0.83)	**2.12**
GA [10]	$46.61_{\pm 6.11}$(53.39)	$49.15_{\pm 9.36}$(50.85)	$99.35_{\pm 0.11}$(0.56)	$95.60_{\pm 0.22}$(0.68)	**26.37**	$0.18_{\pm 0.04}$(3.87)	$2.97_{\pm 1.51}$(3.63)	$99.86_{\pm 0.01}$(0.38)	$97.47_{\pm 0.09}$(1.14)	**2.23**
NegGrad+ [55]	$49.56_{\pm 34.87}$(50.44)	$64.27_{\pm 27.33}$(35.73)	$99.08_{\pm 1.18}$(0.58)	$96.47_{\pm 1.25}$(0.64)	**21.84**	$0.60_{\pm 0.44}$(3.45)	$3.90_{\pm 2.22}$(2.70)	$99.68_{\pm 0.28}$(0.20)	$97.12_{\pm 0.45}$(0.79)	**1.79**
SalUn [5]	$95.13_{\pm 1.79}$(4.87)	$97.24_{\pm 0.17}$(2.76)	$96.33_{\pm 0.25}$(3.33)	$96.18_{\pm 1.10}$(0.93)	**2.97**	$1.15_{\pm 0.40}$(2.90)	$3.56_{\pm 1.12}$(3.04)	$98.97_{\pm 1.56}$(0.51)	$95.42_{\pm 2.10}$(0.91)	**1.84**
SCRUB [19]	$99.12_{\pm 0.14}$(0.88)	$98.01_{\pm 0.51}$(1.99)	$99.75_{\pm 0.03}$(0.09)	$97.24_{\pm 0.16}$(0.13)	**0.77**	$0.18_{\pm 0.08}$(3.87)	$2.80_{\pm 1.33}$(3.80)	$99.88_{\pm 0.03}$(0.40)	$97.36_{\pm 0.15}$(1.03)	**2.28**
Class-F [70]	$71.19_{\pm 3.50}$(28.81)	$63.57_{\pm 4.07}$(36.43)	$61.59_{\pm 4.25}$(38.11)	$59.40_{\pm 4.59}$(37.71)	**35.27**	n/a	n/a	n/a	n/a	n/a
SSD [71]	$91.50_{\pm 0.96}$(8.50)	$91.30_{\pm 8.70}$(8.70)	$99.48_{\pm 0.18}$(0.18)	$97.00_{\pm 0.11}$(0.11)	**4.38**	$0.88_{\pm 0.04}$(3.17)	$5.35_{\pm 0.45}$(1.25)	$99.30_{\pm 0.15}$(0.12)	$96.40_{\pm 0.40}$(0.07)	**1.15**
Ours	$87.23_{\pm 6.55}$(12.77)	$91.41_{\pm 5.90}$(8.59)	$94.77_{\pm 1.16}$(5.14)	$94.04_{\pm 1.29}$(0.88)	**6.85**	$2.27_{\pm 0.50}$(1.78)	$6.13_{\pm 1.40}$(0.47)	$98.29_{\pm 0.32}$(1.19)	$95.82_{\pm 0.48}$(0.51)	**0.99**

In Tab. A1, we provide additional performance overview of different MU methods under two unlearning scenarios on ImageNet-10 using VGG-16. Considering the gain in unlearning effectiveness, the Avg. Gap with Retrain shows that the forget vector remains competitive, except for class-wise forgetting on (ImageNet-10, VGG-16). In this case, SCRUB achieves the best performance, as it leverages model distillation [74], a technique well-suited for class-wise forgetting.

E Component Analysis: λ_1 and λ_2



(a) CIFAR-10, ResNet-18 Class-wise Forgetting (b) CIFAR-10, ResNet-18 Random Data Forgetting (c) ImageNet-10, VGG-16 Class-wise Forgetting (d) ImageNet-10, VGG-16 Random Data Forgetting

Figure A3: Sensitivity analysis of the nonnegative hyper-parameters λ_1 (ranging from 1 to 5 with an interval of 1) and λ_2 (ranging from 0 to 4 with an interval of 1) for image classification under two unlearning scenarios on CIFAR-10 and ImageNet-10 using ResNet-18 and VGG-16, respectively. The unlearning performance is reported using *the average (avg.) performance across UA, RA, TA, and MIA-Efficacy*, with a green star (★) marking the chosen parameter scheme (λ_1 and λ_2) in our experiments. The color bar on the right represents a gradient scale from light to dark red, indicating the range of values (0 to 100%) in the heatmap. The integer within each cell represents the performance (%) given a combination of λ_1 and λ_2.

To verify the impact of each key component in the optimization problem (4) of the main paper, we analyze the nonnegative trade-off parameters λ_1 and λ_2 in Fig. A3. As can be seen in Fig. A3-(a) and

(c), setting the retain loss regularization parameter λ_1 to 3 for CIFAR-10 using ResNet-18 and 5 for ImageNet-10 using VGG-16 in class-wise forgetting, along with the ℓ_2-norm regularization parameter $\lambda_2 = 1$, enables our proposed method to achieve the highest average performance. Meanwhile, for the random data forgetting scenario, setting both λ_1 and λ_2 to 1 yields the best average performance for CIFAR-10 using ResNet-18 and ImageNet-10 using VGG-16.

F Efficiency Analysis

Table A2: The efficiency profile of different MU methods across two metrics for image classification under two unlearning scenarios on CIFAR-10 (RestNet-18) and ImageNet-10 (VGG-16 and ViT). Results are reported in terms of run-time efficiency (RTE) measured in *minutes* for the overall training phase and parameter efficiency denoted by parameter number (Param.#) in Million (M), where a smaller value is favored for each metric.

MU Method	**Class-wise Forgetting**						**Random Data Forgetting**					
	CIFAR-10, ResNet-18		ImageNet-10, VGG-16		ImageNet-10, ViT		CIFAR-10, ResNet-18		ImageNet-10, VGG-16		ImageNet-10, ViT	
	RTE	Param. #	RTE	Param. #	RTE	Param. #	RTE	Param. #	RTE	Param. #	RTE	Param. #
Retrain	81.30	11.17	118.89	15.31	301.98	85.81	103.71	11.17	116.10	15.31	290.60	85.81
FT [4]	4.56	11.17	2.99	15.31	15.40	85.81	2.28	11.17	2.95	15.31	14.47	85.81
RL [2]	5.61	11.17	3.58	15.31	33.00	85.81	4.75	11.17	3.59	15.31	32.84	85.81
GA [10]	1.16	11.17	0.30	15.31	2.10	85.81	1.15	11.17	0.31	15.31	3.10	85.81
NegGrad+ [55]	4.72	11.17	3.21	15.31	16.70	85.81	4.22	11.17	3.17	15.31	16.40	85.81
SalUn [5]	2.81	5.59	2.56	7.66	18.73	42.91	3.01	5.59	2.85	7.66	18.19	42.91
SCRUB [19]	1.23	11.17	1.62	15.31	33.45	85.81	1.53	11.17	1.55	15.31	22.27	85.81
Class-F [70]	1.21	11.17	2.10	15.31	2.86	85.81	n/a	n/a	n/a	n/a	n/a	n/a
SSD [71]	1.20	11.17	2.10	15.31	2.21	85.81	1.15	11.17	2.00	15.31	2.22	85.81
Ours	2.37	0.03	6.15	0.15	4.80	0.15	6.15	0.03	6.50	0.15	8.58	0.15

To demonstrate the efficiency profile of various MU methods under different metrics, we compare the runtime costs of forget vector calculation with those of other model-based unlearning methods, and the total number of updated parameters in the training phase. Specifically, following [5], we use run-time efficiency (RTE) as an evaluation metric, which measures the computation time of applying an MU method in minutes. To ensure a fair comparison, we report the runtime of each approximate MU method within 10 epochs since the number of iterations varies across different MU methods. Additionally, inspired by [75], we compare the number of trainable parameter number (Param.#). Notably, all evaluations are performed in the same computational environment with an NVIDIA A6000 GPU, ensuring fair and reliable comparisons by maintaining a consistent batch size across all MU methods. The corresponding results can be found in Tab. A2, and we can draw the following observation: 1) By comparing the training time of model-based methods with our proposed approach using RTE within the same iterations, we observe that the forget vector method achieves competitive RTE compared to some efficient approximate model-based unlearning baselines and remains faster than retraining. Besides, it is worth noting that the forget vector method optimizes fewer parameters, owing to its input-level design and perturbation space is significantly lower-dimensional. This trade-off underscores the efficiency of our forget vector in handling parameter scaling. For instance, when comparing forgetting cases on ImageNet-10 using VGG-16 and ViT models, we find that as the parameter count increases from "15.31M" in VGG-16 to "85.81M" in ViT, the optimization time for most baseline methods increases significantly. In contrast, our method maintains a consistent efficiency level or even achieves a shorter runtime, highlighting a key advantage of our forget vector approach. 2) From a memory-efficient perspective in real-world application, existing model-based MU methods often necessitate a series of operations (*e.g.*, fine-tuning) on the already-trained model for each forgetting request. This necessitates storing a separate model version for every request, leading to substantial storage overhead. In contrast, our approach optimizes an input-level universal perturbation, where only the "forget vector" associated with the data to be forgotten is stored with original model remains intact. Since each forget vector has the same dimensionality as the input image (*e.g.*, 0.03M for CIFAR-10 and 0.15M for ImageNet) and is significantly smaller than the full model, our approach offers a considerable advantage in storage efficiency for practical applications.

G Additional Visualization through An Input Saliency Lens

In Figs. A4 and A5, we provide additional visualization results through input saliency map for different methods against forget images and retain images, respectively, using Grad-CAM (Gradient-weighted Class Activation Mapping) [72]. Consistent with the main paper, we highlight the salient pixels (*i.e.*, regions most influential to model predictions) under four scenarios: (1) the original model (without unlearning), (2) Retrain, (3) SCRUB-based unlearning, and (4) forget vector-based unlearning. For the first three scenarios, saliency maps are generated on raw forget/retain images, whereas for the last scenario, they are applied to images perturbed by the forget vector.

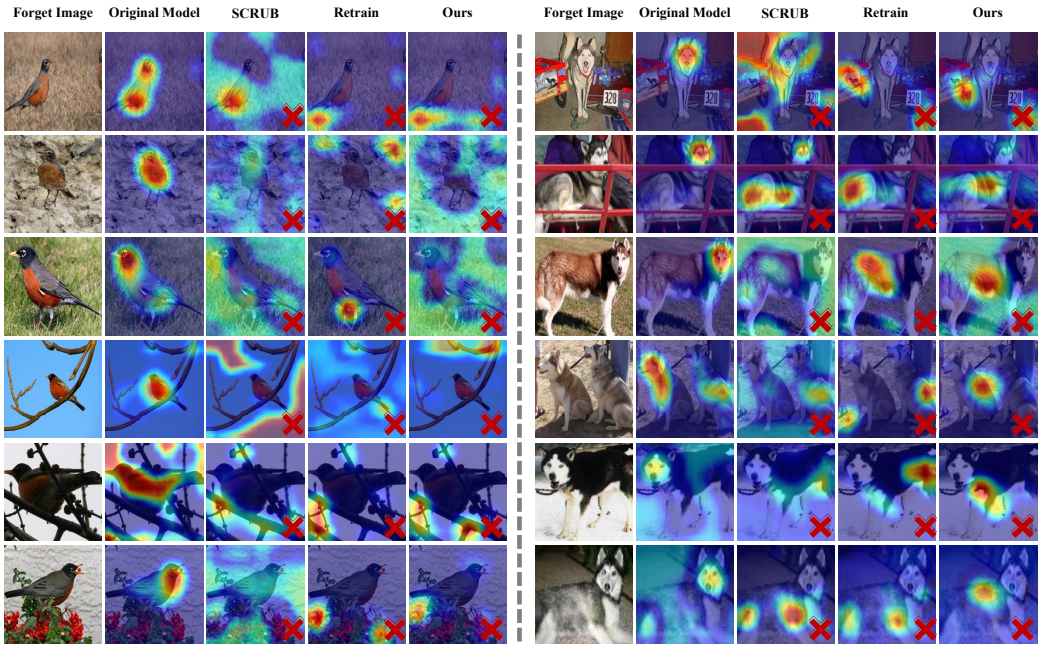


Figure A4: Gradient-based saliency map visualized via Grad-CAM for different MU methods against **forget images**. The highlighted areas (marked in red) indicate regions most influential to model prediction, and the red cross mark (✖) indicates that corresponding methods effectively unlearn the input forget images.

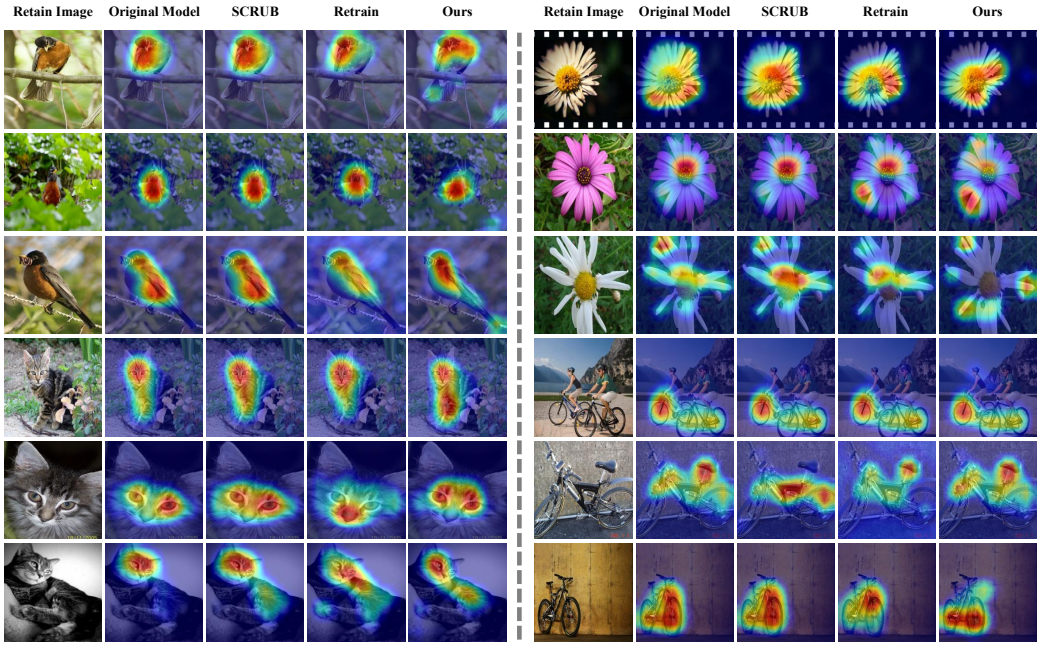


Figure A5: Gradient-based saliency map visualization using Grad-CAM for different MU methods against **retain images**.

H Broader Impacts and limitations

Broader impacts. Our study on the forget vector introduces a novel, data-driven machine unlearning approach that offers significant potential across various domains. ① Privacy Preservation and Regulatory Compliance: With increasing global regulations like GDPR mandating the right to be forgotten, our method enables effective unlearning of specific data points without requiring full model retraining, which is especially valuable for industries handling sensitive data, such as healthcare,

finance, and personalized recommendation systems. By enabling compliant data removal with minimal computational overhead, our approach strengthens data privacy practices while maintaining model integrity. ② Adaptability Across Tasks: The compositional nature of forget vectors allows for the unlearning of previously unseen tasks by combining class-specific forget vectors. This adaptability extends the method's applicability, where our method is able to respond more quickly when faced with new random unlearning tasks. ③ Efficiency in Unlearning and Model Storage: Unlike model-based MU approaches that require significant computational resources and model storage for each unlearning request, our method provides an energy-efficient alternative with less trainable parameters and low storage of optimized forget vector. This efficiency is particularly beneficial for organizations deploying large-scale AI models, where frequent updates or compliance-driven data deletions could otherwise be prohibitively expensive, contributing to the sustainability of AI development.

Limitations. While our forget vector approach presents compelling advantages, it also comes with certain limitations that need to be addressed for broader adoption: ① Vulnerability to Adversarial Attacks: Since our method relies on input perturbations rather than direct model modifications, it may be vulnerable to white-box adversarial attacks where an attacker has access to the model and understands the forget vector mechanism. These adversaries could potentially reconstruct forgotten data or design countermeasures to bypass unlearning. Future work should focus on strengthening robustness against such attacks through adversarially hardened perturbation strategies. ② Computational Cost of Compositional Unlearning: The generation of new forget vectors through compositional operations requires access to pre-trained class-wise forget vectors. In scenarios where these vectors are unavailable, applying our method at scale could introduce overhead. ③ Lack of Theoretical Guarantees: While empirical results demonstrate the effectiveness of our method, formal theoretical guarantees on its unlearning performance and robustness remain an open challenge. Future work should focus on establishing rigorous mathematical foundations for the forget vector framework. ④ Generalization to Other Domains: Our current study focuses on image classification, and its applicability to other domains, such as natural language processing or image generation, remains to be explored. Investigating the feasibility of forget vectors in diverse machine learning tasks is an important direction for future research.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly outlines the key contributions, and appropriately limits its scope to machine unlearning task.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations discussion that need to be addressed for broader adoption can be found in Appendix H.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The paper provides a complete set of assumptions and correct proofs for each theoretical result as detailed in sections 4 and 5.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper includes comprehensive information and clear guidelines for reproducing the main experimental results, covering model training, parameter selection, and evaluation protocols, as outlined in sections 5 and 6.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The complete source code, along with detailed instructions for running the code and the required environment setup have been included in the supplementary material. The code is designed to work with publicly available datasets. For the ImageNet10 dataset, we have clearly listed the specific categories used in the supplementary material, ensuring the faithful reproduction of the main experimental results.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Comprehensive details regarding the training and test settings, including data splits, hyperparameter configurations, optimizer choices are included in section 6 and Appendix A, C, E, ensuring that the experimental setup is fully reproducible and the results are accurately interpretable.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The paper accurately reports statistical significance by including the mean and standard deviation values computed from 10 independent trials for each method (*e.g.*, Table 1, Table 2, Table 3, and Figure 2). This approach provides a robust measure of performance variability, clearly reflecting the consistency and reliability of the experimental results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper provides detailed information on the computational resources required for each experiment, including the runtime efficiency (RTE) in minutes and parameter numbers (Param. #) in millions, as presented in Table A2 in Appendix F.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in this paper fully complies with the NeurIPS Code of Ethics. The experiments use publicly available datasets, and all necessary measures have been taken to preserve participant privacy and data anonymity. No personal or sensitive information was used, and the study adheres to ethical standards for data handling and model evaluation.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: The broader impacts of our work are discussed in detail in Appendix H, where we outline the potential positive impacts, including privacy preservation, regulatory compliance, adaptability across tasks, and efficiency in unlearning and model storage. These discussions highlight the societal benefits of our proposed approach, particularly in contexts requiring efficient and scalable machine unlearning.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [Yes]

Justification: The paper carefully considers the potential risks associated with model misuse and has implemented appropriate safeguards for responsible release. Specifically, the models used in this work are trained on publicly available datasets with clearly defined and ethically compliant usage licenses. Additionally, the paper explicitly addresses privacy concerns through the use of forget vectors, which ensure sensitive data can be effectively unlearned, reducing the risk of unintended data retention or misuse.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Detailed citations are included in the main text and Appendix to ensure proper acknowledgment of the original contributors. The license terms and conditions for these assets adheres to the respective open-source licenses.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The source code is well-documented in the supplementary materials, where comprehensive instructions for usage are included to facilitate reproducibility.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing or research with human subjects. All experiments are conducted using publicly available datasets and machine learning models, without the need for human participant data.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve research with human subjects and therefore does not require Institutional Review Board (IRB) approval or equivalent ethical review. All experiments were conducted using publicly available datasets without direct human involvement.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core methods and contributions of this paper do not involve the use of large language models (LLMs) as an essential, original, or non-standard component.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.