FF-ERASE: MACHINE UNLEARNING AND VERIFICATION FOR FORWARD-FORWARD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

The Forward-Forward (FF) algorithms provide a promising alternative to backpropagation (BP) by training neural networks through independent layer-wise goodness objectives. However, machine unlearning for FF models remains largely unexplored, which is a fundamental model function aiming to remove the influence of designated training samples from a trained model while preserving its utility on the remaining data. Directly adapting BP-oriented unlearning methods either causes ineffective unlearning or induces model collapse. This is because of two fundamental challenges specific to FF unlearning: (i) each FF layer lacks a compatible unlearning target with the other layers (*i.e.*, coordinated yet layer-specific directions), and (ii) the under/over-forgetting balance is independent across layers and thereby difficult to determine. To address these challenges, we propose FF-Erase, the first FF-specific machine unlearning framework. FF-Erase leverages a low-cost guidance model to instantiate a stable and structurally realizable goodness space. This establishes coordinated yet layer-specific anchors for precise goodness realignment, thereby avoiding the optimization collapse of independent layers. Such a goodness anchor also enables layer-independent unlearning adaptation, allowing FF-Erase to adjust the under/over-forgetting balance layer-wise. We further propose goodness-based membership inference (G-MIA), an FF-specific unlearning auditing tool that efficiently and accurately verifies the removal effectiveness for an FF unlearning algorithm. Extensive experiments across architectures, datasets, forgetting ratios, and unlearning scenarios show that FF-Erase consistently forgets target data effectively while preserving retained-data utility and avoiding model collapse. FF-Erase achieves up to $11.2\times$ speedup over retraining, establishing an effective and efficient foundation for machine unlearning on FF models.

1 INTRODUCTION

The Forward-Forward (FF) algorithms (Hinton, 2022) have emerged as a promising alternative to backpropagation (BP) for training deep learning models. This approach updates the parameters of each layer independently by greedily optimizing a layer-wise goodness objective derived from its activations. By maximizing this score for positive data (*i.e.*, correctly labeled data) and minimizing it for negative data (*e.g.*, invalid or incorrectly labeled data) during the forward pass, FF algorithms can train model parameters without requiring a backward pass that blocks all layers. This BP-free nature is considered more biologically plausible and brings significant practical advantages, including reduced memory overhead from stored activations and efficient training using pipeline parallelism. These features make FF particularly suitable for resource-constrained scenarios like edge computing.

However, the critical task of machine unlearning for FF models remains a foundational yet unexplored problem. First, edge devices (*e.g.*, wearables and medical sensors) inherently process highly sensitive and user-specific data. The data owners demand the "right to be forgotten" (RTBF), which has been explicitly stated in the EU General Data Protection Regulation (GDPR)(Voigt & Von dem Bussche, 2017) and the California Consumer Privacy Act (CCPA)(Harding et al., 2019). Beyond privacy, the model also needs to unlearn outdated, poisoned, or mislabeled data for better model reliability(Wang et al., 2025; Zhang et al., 2023). Therefore, it is essential to enable FF machine unlearning, which aims to remove the influence of specific training samples from a trained model (*i.e.*, effectiveness) while preserving its performance on the remaining data (*i.e.*, model utility).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Existing machine unlearning methods are primarily designed for BP models whose parameters are coordinated by a global unlearning objective, and therefore not directly applicable to FF models. The most straightforward approach, retraining the model from scratch on the remaining data, is computationally prohibitive. Approximate methods based on gradient ascent, projection, or distillation are incompatible with FF's layer-wise optimization. As illustrated in Figure 1, naively adapting these methods to independently optimized FF layers exposes two FF-specific challenges.

Firstly, FF models exhibit severe optimization instability under unlearning. This is because their independently optimized layers lack a jointly realizable set of layer-wise targets. In BP, layer-wise updates are jointly constrained by a common global objective through the chain rule. In contrast, FF layers greedily optimize separate local goodness objectives. Merely penalizing the goodness of forgotten data dictates which representations a layer should discard, but fails to provide an empirically validated target state for it to converge upon. Therefore, different layers risk drifting toward mutually incompatible goodness states that destabilize the unlearning and cause model collapse. Conversely, forcing all layers toward the same artificial target is also inappropriate because heterogeneous FF layers naturally encode different representations. Effective FF unlearning thus requires coordinated yet layer-specific goodness directions that are empirically supported by a stable FF model.

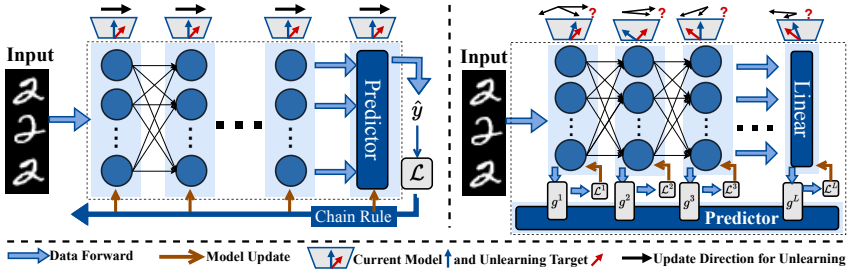


Figure 1: Why BP-oriented unlearning does not directly transfer to FF models. In BP models (left), a global objective coordinates layer-wise updates through the chain rule. In FF models (right), independently optimized local objectives lack jointly compatible layer-wise targets and require heterogeneous unlearning strengths, leading to under-forgetting, over-forgetting, or model collapse.

Secondly, besides which direction to unlearn in, it is also challenging for FF models to determine how much each layer should forget. Different layers encode different levels of abstraction and exhibit heterogeneous local under/over-forgetting balance. For example, early CNN layers primarily capture generic patterns such as edges and textures, whereas deeper layers encode more class-specific features (Yosinski et al., 2014). In BP, a shared global objective and the chain rule jointly determine the forgetting strength across layers. FF layers lack a coordination mechanism to globally determine this strength. Consequently, given the same forgetting request, some layers are already over-forgetting while other layers are under-forgetting. If the unlearning continues, some layers are risking collapse; otherwise, the remaining layers still contain the knowledge of target data and are thereby ineffective.

The above discussion motivates our first key question: *How can we provide appropriate layer-wise unlearning directions and adapt the under/over-forgetting balance across heterogeneous FF layers?*

Moreover, efficiently verifying FF unlearning performance is non-trivial. MIAs are widely used as empirical unlearning audits (Gao et al., 2024), but high-performing attacks often introduce substantial verification overhead. Shadow-model-based attacks require additional model training (Liu et al., 2022a; Carlini et al., 2022; Zarifzadeh et al., 2023); white-box attacks consume large intermediate features or gradients (Leino & Fredrikson, 2020; Cohen & Giryes, 2024; Suri et al., 2024; Golbari et al., 2025); and perturbation-based attacks require repeated model queries (Chen et al., 2022; Jayaraman et al., 2020; Ali et al., 2023; Ravikumar et al., 2024). In our FF setting, these costs can approach or exceed the unlearning cost itself. This motivates our second key question: *How can FF-specific information be exploited for accurate yet lightweight unlearning verification?*

To address these questions, we make the following contributions:

- *Problem Formulation and FF-Specific Challenges*: To the best of our knowledge, we present the first study of machine unlearning for FF models. We identify two fundamental challenges introduced by independent layer-wise optimization: (i) the lack of coordinated yet layer-specific

directions for goodness realignment, and (ii) heterogeneous local under/over-forgetting balance that cannot be handled by a single global adjustment strength.

- *Guidance-Driven FF Unlearning*: We propose FF-Erase, which efficiently unlearns the target data through precise goodness realignment. FF-Erase leverages a low-cost guidance model to construct coordinated yet layer-specific goodness anchors, which effectively addresses the FF-specific challenges and erases the data influence on FF models without risking optimization collapse.
- *Goodness-Based Unlearning Verification*: We propose goodness-based membership inference (G-MIA), an FF-specific unlearning verification tool that leverages the layer-wise goodness outputs from FF layers. G-MIA requires neither shadow-model training, repeated perturbation queries, nor large intermediate features and gradients, and achieves the highest mean ACC/AUC among the evaluated MIA baselines with substantially lower verification overhead.
- *Extensive Evaluation*: We evaluate FF-Erase across various FF architectures, datasets, forgetting ratios, and unlearning scenarios, including IID, class-wise, backdoor, mislabeled, non-IID, and high-influence unlearning. The results show that FF-Erase consistently realizes effective removal while preserving retained-data utility, achieving up to $11.2\times$ speedup over retraining.

2 Related Work

Machine Unlearning: Retraining provides the gold standard of unlearning effectiveness and model utility, but it is costly. Approximate unlearning accelerates removal by directly modifying the trained model. SOTA approaches include gradient-based methods (Tarun et al., 2023; Sekhari et al., 2021), projection-based methods such as PGU (Hoang et al., 2024), distillation-based methods (Kurmanji et al., 2023; Chundawat et al., 2023), reference-guided distribution matching such as ReGUn (Mirlach et al., 2026), and influence-based parameter updates (Qiao et al., 2024; Liu et al., 2022b). These methods are designed around globally coupled BP optimization and do not explicitly provide corresponding-layer FF goodness anchors or adapt the relative objective scale to each FF layer. FF-Erase addresses these two issues through model-generated layer-wise targets and gradient-magnitude balancing, realizing the first effective and efficient FF unlearning.

Unlearning Verification: Membership inference attacks are widely used for empirical unlearning verification (Gao et al., 2024; Du et al., 2024). The goal of an MIA is to determine if a given sample was in a model's training set. If an unlearning method is effective, MIAs should not successfully infer the forgetting samples as members. Literature improves their attack accuracy for more reliable removal verification, but their efficiency shortcomings also make them inefficient for FF models. Shadow-model-based attacks improve attack power through additional reference models (Liu et al., 2022a; Carlini et al., 2022; Zarifzadeh et al., 2023), but incur substantial training cost. White-box attacks exploit intermediate activations, parameters, or gradients (Leino & Fredrikson, 2020; Cohen & Giryes, 2024; Suri et al., 2024; Golbari et al., 2025), while perturbation-based attacks require repeated target-model queries (Chen et al., 2022; Jayaraman et al., 2020; Ali et al., 2023; Ravikumar et al., 2024). These approaches provide strong auditing signals but can be costly when applied to FF models. G-MIA instead exploits the compact goodness vectors already produced by FF layers, providing an architecture-aware, one-shot auditing interface with substantially lower overhead.

3 Preliminaries

Forward-Forward Algorithm and its Development: Hinton (2022) proposed the original FF model as a two-forward binary goodness discrimination. It explicitly requires positive data (real data with correct label) and negative data (invalid data or data with the wrong label). Then the layer-wise goodness is defined as the squared L2 norm of the layer output. For positive (negative) forward, parameters at layer l are optimized to increase (decrease) the positive goodness g_+^l (negative goodness g_-^l), where goodness g_+^l and g_-^l are scalar values. To stabilize the FF training, Lee & Song (2023) jointly optimized the positive and negative goodness values through a symmetric contrastive loss, which enlarges the difference between two goodness values $g_+^l - g_-^l$. Papachristodoulou et al. (2024) extend FF to multi-class classification, changing goodness from scalars into vectors. Consider an FF model containing L layers for a J-class classification task. Firstly, the goodness at layer l becomes a vector $\boldsymbol{g}^l = [g_1^l, g_2^l, \ldots, g_J^l]$. Let $\boldsymbol{x}$ denote the input data and y its correct label. While g_y^l is positive goodness, the residual $J-1$ scores $g_{\neq y}^l$ are negative goodness. Therefore, the negative examples are

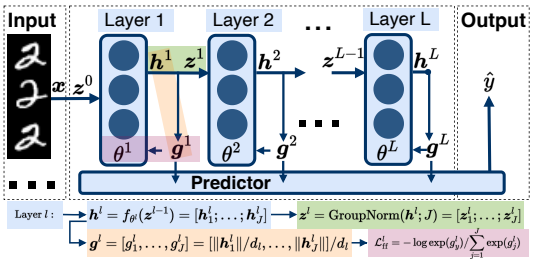


Figure 2: Illustration of the multi-class FF backbone and its layer-wise training. After each layer computes pre-normalization activations $\boldsymbol{h}^l$, it in parallel forwards their normalized representation $\boldsymbol{z}^l$ to the next layer and constructs the class-wise goodness vector $\boldsymbol{g}^l$ for its local parameter optimization.

not required explicitly, and we update the parameters using cross-entropy loss, which automatically enlarges the difference between positive and negative goodness. Sun et al. (2025); Xu et al. (2026); Torres et al. (2025); Gautham et al. (2024); Park et al. (2024) further extend FF to deeper models and various scenarios, but the common backbone above is widely accepted and we formalize it as follows.

Notations of FF Backbone:[1] As illustrated in Figure 2, let f_{θ^l} denote the function of layer l before normalization, parameterized by θ^l. Starting from $\boldsymbol{z}^0 = \boldsymbol{x}$, each layer computes $\boldsymbol{h}^l = f_{\theta^l}(\boldsymbol{z}^{l-1}) = [\boldsymbol{h}^l_1; \ldots; \boldsymbol{h}^l_J]$, where $\boldsymbol{h}^l$ denotes the pre-normalization activations partitioned into J class-associated feature groups. Next, the activations are normalized class-wise before being forwarded to the next layer: $\boldsymbol{z}^l = \mathrm{GroupNorm}(\boldsymbol{h}^l; J) = [\boldsymbol{z}^l_1; \boldsymbol{z}^l_2; \ldots; \boldsymbol{z}^l_J]$. In parallel, the goodness vector at layer l is defined as $\boldsymbol{g}^l = [g^l_1, g^l_2, \ldots, g^l_J]$, where each scalar goodness score is computed from its corresponding feature group as $g^l_i = \|\boldsymbol{h}^l_i\|_1 / d_l$, with d_l denoting the number of activations in each feature group $\boldsymbol{h}^l_i$.[2] The model aggregates layer-wise outputs for prediction, *i.e.*, $\hat{y} = f_{\mathrm{predictor}}(\boldsymbol{g}^1, \boldsymbol{g}^2, \ldots, \boldsymbol{g}^L)$.

Training FF Models: Given the input $(\boldsymbol{x}, y)$, FF training encourages the goodness score $g^l_{i=y}$ of the correct class (*positive*) to dominate the goodness scores of all incorrect classes (*negative*) $g^l_{j\neq y}$. Following Papachristodoulou et al. (2024); Sun et al. (2025), the training process is as follows:

$$\theta^l \leftarrow \theta^l - \eta \nabla_{\theta^l} \mathcal{L}^l_{\mathrm{ff}}(\boldsymbol{g}^l, y), \qquad \mathcal{L}^l_{\mathrm{ff}}(\boldsymbol{g}^l, y) = \mathcal{L}_{\mathrm{CE}}(\mathrm{softmax}(\boldsymbol{g}^l), \boldsymbol{y}) = -\log \frac{\exp(g^l_y)}{\sum_{j=1}^{J} \exp(g^l_j)}, \quad (1)$$

where $l = 1, 2, \ldots, L$, $\boldsymbol{y}$ is the one-hot vector of label y and η is the learning rate.

Goodness distribution: We define the *goodness distribution* as the relative weighting of the class-wise goodness scores in $\boldsymbol{g}^l$, computed as $\mathrm{softmax}(\boldsymbol{g}^l)$. We formulate the process of driving this distribution toward a specific target distribution (*e.g.*, by minimizing their KL divergence) as *goodness alignment*. Under this definition, standard training is essentially aligning the goodness distribution with the ground-truth one-hot vector of y, which inherently enlarges the margin $g^l_y - g^l_{j\neq y}$. In the context of unlearning, one might intuitively attempt to align the goodness distribution with a one-hot vector of an incorrect label or a flattened uniform distribution. However, not every arbitrary goodness vector corresponds to valid and realizable representation states for an FF model. Forcing independent layers to align with such invalid goodness targets risks severe model collapse, as demonstrated empirically in §6.3. This explains why it is important to answer where to unlearn in FF unlearning.

4 METHODOLOGY

Machine Unlearning Objective: The goal of a machine unlearning process is to remove the influence of forgetting data $\mathbb{D}_{\mathrm{f}}$ from an original model θ_o (the model to be unlearned) while preserving the utility of the unlearned model θ_u on the remaining data $\mathbb{D}_{\mathrm{r}} = \mathbb{D}_{\mathrm{train}} \setminus \mathbb{D}_{\mathrm{f}}$, where $\mathbb{D}_{\mathrm{train}}$ is the training

[1] For concreteness, we formalize the FF backbone following the representative multi-class FF architecture in Sun et al. (2025). Our FF-Erase also applies to the original FF setting, as evaluated in Appendix §A.

[2] Unless otherwise stated, the CNN experiments use the mean-activation goodness in Sun et al. (2025); the original FF-MLP experiment follows the sum-of-squares goodness definition in Hinton (2022).

dataset of θ_o. This unlearning objective can be formalized as the combination of the two terms below:

$$\min_{\theta_u \in \Theta} \mathcal{L}_{\text{removal}} + \lambda \mathcal{L}_{\text{utility}}, \tag{2}$$

where λ controls the trade-off between the effectiveness (*i.e.*, $\mathcal{L}_{\text{removal}}$) and the utility (*i.e.*, $\mathcal{L}_{\text{utility}}$).

FF-Erase Workflow: As established in §3, FF-Erase aims to search for a goodness anchor, which is not only valid for FF models (to avoid collapse caused by misalignment) but also ignorant of target knowledge (to guide unlearning). Moreover, this goodness anchor should be layer-wise compatible, as required by the Challenge (i) in §1. To this end, FF-Erase uses a guidance model θ_g to provide an empirically valid set of layer-specific goodness anchors. The guidance model θ_g is a frozen model of the same architecture as θ_o, which provides FF-Erase with an anchor $\boldsymbol{g}_*^l$ for each layer by forwarding $\boldsymbol{x}$ through θ_g. We initialize the unlearned model from θ_o and, for each forgetting sample, align its layer-wise goodness distributions toward the corresponding guidance anchors:

$$\mathcal{L}^l_{\text{removal}} = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathbb{D}_\text{f}} D_{\text{KL}}\big(\text{softmax}\big(\boldsymbol{g}_*^l(\boldsymbol{x};\theta_g)\big) \,\|\text{softmax}\big(\boldsymbol{g}_u^l(\boldsymbol{x};\theta_u)\big)\big)\,. \tag{3}$$

Importantly, the guidance model does not define what makes a goodness distribution valid; rather, its corresponding-layer outputs provide an observable, jointly compatible instance from the structured goodness space of a stable FF model. This gives different FF layers coordinated yet layer-specific destinations instead of forcing them toward an identical artificial target. The utility loss follows the FF objective on remaining data $\mathbb{D}_\text{r}$, *i.e.*, $\mathcal{L}^l_{\text{utility}} = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathbb{D}_\text{r}} \mathcal{L}^l_{\text{ff}}\big(\boldsymbol{g}_u^l(\boldsymbol{x}), y\big)$. Therefore, the overall unlearning process of $\mathcal{L}_{\text{FF}-\text{Erase}}$ is layer-wise and is formulated as follows:

$$\forall l \in \{1,\ldots,L\}, \quad \theta_u^l \leftarrow \theta_u^l - \eta \nabla_{\theta_u^l}\left(\mathcal{L}^l_{\text{removal}} + \lambda^l \mathcal{L}^l_{\text{utility}}\right). \tag{4}$$

At each unlearning update step, we determine the utility weight directly from the current layer-wise gradient magnitudes:$\lambda^l = \|\nabla_{\theta_u^l}\mathcal{L}^l_{\text{removal}}\|_2/(\|\nabla_{\theta_u^l}\mathcal{L}^l_{\text{utility}}\|_2 + \epsilon_{\text{num}})$, where ϵ_{num} is a small constant preventing division by zero. The λ^l is first computed from the current layer-wise gradient norms and then fixed as a scalar during parameter updating. Such balancing techniques have been widely explored in multi-objective optimization (Chen et al., 2018; He et al., 2022), and we extend them to the FF scenario for unlearning purposes. This extension works together with goodness anchors, allowing each layer to independently adjust its goodness realignment according to its own unlearning target and strength. Therefore, the under-forgetting layers can continue to minimize removal loss while over-forgetting layers turn to repair utility, thereby addressing Challenge (ii) in §1.

Validation-Trained (Val-Train) Guidance Model. In standard ML pipelines with train/val/test splits, the validation set $\mathbb{D}_\text{v}$ is already defined and disjoint from the training set $\mathbb{D}_{\text{train}}$. Our Val-Train method leverages this existing $\mathbb{D}_\text{v}$ to train a fresh FF guidance model θ_g to convergence. Since $\mathbb{D}_\text{v}$ typically constitutes only a marginal fraction of the overall dataset, obtaining θ_g only incurs minimal computational overhead compared to full retraining.[3] It naturally provides valid goodness distribution as a converged model, and it is ignorant of target knowledge as its training set is disjoint from $\mathbb{D}_{\text{train}}$.

5 Goodness-based Membership Inference

G-MIA Workflow: We train a binary classifier for score-based membership inference, leveraging the FF-specific information contained in the layer-wise goodness outputs. We formalize it as follows:

$$f_{\text{G}-\text{MIA}}(\widetilde{\boldsymbol{g}}^1, \widetilde{\boldsymbol{g}}^2, \ldots, \widetilde{\boldsymbol{g}}^L) = \begin{cases} 1, & \textit{member} \\ 0, & \textit{non-member} \end{cases}, \text{ where } \widetilde{\boldsymbol{g}}^l = \text{softmax}\left(\frac{\boldsymbol{g}^l}{T}\right) \tag{5}$$

Here T is a constant to sharpen the goodness distribution as detailed in Appendix C.3.

Audit Access Model. G-MIA is designed for the model owner or an authorized auditor acting on behalf of data owners, rather than as a common malicious attack under strict black-box assumptions. We aim for accurate and efficient unlearning verification. Therefore, the lightweight layer-wise goodness vectors (usually within 1KB) are accessible for G-MIA. G-MIA does not require intermediate activations or gradient behaviors for efficiency needs.

[3]Usually, the $\mathbb{D}_\text{v}$ set is split from 10% of the raw training data. It only takes around 5% of retraining time to obtain this θ_g as shown in §6.3, where we further show Val-Train's robustness with a split ratio across 1%-10%. It is noted that FF-Erase does not rely on Val-Train; there are alternatives to obtain θ_g as shown in §6.3 and Appendix §B. We use it by default because it is recyclable for sequential unlearning and it is efficient to obtain.

Unlearning Verification: For unlearning verification, we train the G-MIA classifier on the original model θ_o. We then freeze the attack model and apply G-MIA to the unlearned FF model θ_u on a balanced verification set containing equal numbers of $\mathbb{D}_{\text{forget}}$ samples and $\mathbb{D}_{\text{test}}$ samples. If G-MIA ACC/AUC drops toward 0.5, it shows that G-MIA no longer distinguishes forgotten samples from genuinely unseen samples. A lower G-MIA score indicates a more effective unlearning process.

Experiments on G-MIA We compare G-MIA against SOTA MIA methods using an FF VGG-13 model and CIFAR-10 dataset. For SOTA shadow-model-based MIAs, we use FL (Liu et al., 2022a), RMIA (Zarifzadeh et al., 2023), and LiRA (Carlini et al., 2022) as baselines, where FL and RMIA train one shadow model and LiRA requires 64 shadow models. AMIA (Ali et al., 2023) and CurvMIA (Ravikumar et al., 2024) are SOTA boundary-perturbation MIA methods, where AMIA requires 200 queries per sample, and CurvMIA needs 50 queries per sample. Grad-MIA (Nasr et al., 2019), IHA (Suri et al., 2024) and ImpMIA (Golbari et al., 2025) are the SOTA white-box MIAs.

Table 1: Comparison among MIAs on the ACC, AUC, input size d_{in}, time for shadow models $t_{\text{sh.m}}$ and attacking time for one data sample t_{mia}. We bold the **best** result and underline the second-best result for each row. Due to space limits, results on other benchmarks are provided in Appendix §D.

Metric	G-MIA	FL	RMIA	LiRA	AMIA	CurvMIA	Grad-MIA	IHA	ImpMIA
ACC ↑	**0.6125**	0.5415	0.5708	0.5919	0.5687	0.5793	0.5951	0.6035	0.6021
AUC ↑	**0.6315**	0.5662	0.5633	0.6127	0.5882	0.6135	0.6178	0.6238	0.6269
d_{in} ↓	0.52KB	**0.04KB**	**0.04KB**	**0.04KB**	**0.04KB**	**0.04KB**	458.8MB	408.8MB	408.1MB
$t_{\text{sh.m}}$ ↓	**0s**	986.5s	983.6s	63104.3s	**0s**	**0s**	**0s**	**0s**	**0s**
t_{mia} ↓	**6ms**	**6ms**	11 ms	0.63s	8.62s	4.18s	2.20s	8.85s	8.78s

As shown in Table 1, experiments demonstrate that G-MIA achieves the highest accuracy (ACC and AUC) for FF models. Meanwhile, FL, RMIA, and LiRA require infeasible time for training shadow models ($t_{\text{sh.m}}$ ranging from 983s to over 63,104.3s, as costly as retraining RE in §6.1). Boundary-perturbation methods (AMIA and CurvMIA) avoid shadow models but require impractically long t_{mia} per sample (G-MIA is 4.18s/6ms = 696.7× faster) due to their multi-query nature. Furthermore, while white-box methods (Grad-MIA, IHA, ImpMIA) achieve competitive accuracy, they demand impractically massive input sizes ($d_{\text{in}} > 400$ MB), leading to prohibitive GPU memory costs. In contrast, G-MIA successfully bridges this gap. It requires zero shadow-model training ($t_{\text{sh.m}} = 0$), maintains a negligible input footprint ($d_{\text{in}} = 0.52$ KB), and executes exceptionally fast ($t_{\text{mia}} = 6$ ms). Thus, among the evaluated attacks, G-MIA achieves the highest mean ACC/AUC while requiring no shadow-model training, only 0.52 KB of goodness inputs on VGG-13/CIFAR-10, and approximately 6 ms per inference. Due to space limits, complete results are deferred in Appendix §D, which shows consistent accuracy-efficiency advantages of G-MIA compared with baseline MIAs on other datasets.

Advantage analysis: We attribute the success of G-MIA to its utilization of layer-wise information unique to FF models. To validate this, we conduct an ablation study (Table 2) evaluating G-MIA with different combinations of goodness vectors. Specifically, we compare using only the final layer, the best single layer (we test every layer and find that G-MIA on layer 6 is the best), uniformly distributed subsets of four (Layers 1, 5, 9, 13) and seven layers (Layers 1, 3, 5, 7, 9, 11, 13), and all 13 layers.

Table 2: ACC/AUC of G-MIA using different combinations of goodness vectors. We repeat over five random seeds and report the mean and relative standard deviation (RSD), $\text{RSD} = 100\sigma/|\mu|\%$.

Setting	Metric	Final Layer	Best Single Layer	Four Layers	Seven Layers	ALL Layers	FL 2022a
VGG-13	ACC↑	$0.5303_{\pm 0.4\%}$	$0.5808_{\pm 0.6\%}$	$0.5813_{\pm 0.4\%}$	$0.5982_{\pm 0.9\%}$	$0.6125_{\pm 0.7\%}$	$0.5415_{\pm 0.6\%}$
CIFAR-10	AUC↑	$0.5651_{\pm 0.5\%}$	$0.6018_{\pm 0.5\%}$	$0.6054_{\pm 0.6\%}$	$0.6179_{\pm 0.8\%}$	$0.6315_{\pm 0.7\%}$	$0.5662_{\pm 0.7\%}$

Table 2 shows that membership information in FF models is broadly distributed across layers rather than concentrated in the final output. Relying solely on final-layer goodness yields an ACC/AUC of 0.5303/0.5651, nearly identical to the output-based FL attack (0.5415/0.5662) and significantly lower than the best result of using another single layer (0.5808/0.6018). More importantly, incorporating progressively more layers consistently strengthens G-MIA: the AUC increases from 0.6054 (four layers) to 0.6179 (seven layers) and peaks at 0.6315 (all layers). Overall, aggregating all layer-wise goodness improves ACC/AUC by 8.22/6.64 percentage points over the final layer alone. This confirms that FF layers expose complementary membership information, aligning with FF's independent layer-wise training objective that *knowledge of each layer is not necessarily propagated to the final output.*

6 EVALUATION

Experimental settings. We evaluate FF unlearning using the representative VGG-13/CIFAR-10 setting in the main text; additional architectures and datasets, including the original FF-MLP, ResNet-50, MNIST, Fashion-MNIST, and CIFAR-100, are reported in the appendix. All experiments are implemented in PyTorch and executed on NVIDIA A800 GPUs. Each experiment is repeated over five random seeds. Table subscripts report the relative standard deviation (RSD), $\mathrm{RSD} = 100\sigma/|\mu|\%$. For small RSDs and constants, we report the equivalent value in ‰ and 0.00 for readability, respectively. In the following section, we will first show FF-Erase's unlearning performance in §6.1. Next, in §6.2, we summarize the experimental results that are deferred in the appendix due to space limits. Finally, in §6.3, we show the ablation study on FF-Erase's design and guidance model choices.

6.1 MACHINE UNLEARNING ON FF MODELS

Baselines. We compare FF-Erase against representative unlearning methods spanning gradient-based methods GA (Sekhari et al., 2021) and FYE (Tarun et al., 2023), the projection-based method PGU (Hoang et al., 2024), distillation-based methods SCRUB (Kurmanji et al., 2023) and Bad Teacher (BT) (Chundawat et al., 2023), the reference-guided method ReGUn (Mirlach et al., 2026), and the retraining-based method SISA (Bourtoule et al., 2021). For these BP-oriented approximate methods, we adapt them in two ways in evaluation: (1) direct adaptation, which applies the original method to FF models, and (2) layer-wise adaptation, which repeats the corresponding objective independently for every FF layer. For layer-wise adaptation, we explicitly report them as (layer-wise)+Method.

Unlearning Setting and Metrics. Here, we randomly select 20% samples from training set as the forget set $\mathbb{D}_\mathrm{f}$ for unlearning. For comprehensive assessment, we consider the following metrics: (i) **efficiency**, measured by overall unlearning time $\mathrm{t}_{\mathrm{total}}$ (for θ_o, it refers to training time); (ii) **effectiveness**, evaluated via G-MIA AUC and accuracy on the forgetting set (Acc_f); and (iii) **utility**, using accuracy on the test set (Acc_t). We also report retraining (RE) as the baseline for assessing Acc_f and Acc_t results: the closer they are to RE, the better the unlearning method performs.

As shown in Table 3, FF-Erase achieves the strongest effectiveness–utility trade-off among efficient methods without model collapse. It completes unlearning in 139.7 s, including 57.1 s for training the guidance model and 82.6 s for FF updating, corresponding to a 7.9× speedup over RE. Its forgetting-set accuracy is 81.95%, with G-MIA = 0.5135 and retained test accuracy = 79.48%, all close to the corresponding RE behavior. SISA provides effective removal but requires nearly the full retraining cost and retains lower utility. Direct adaptations of GA, FYE, PGU, SCRUB, BT, and ReGUn largely preserve utility but leave substantial residual information: their forgetting-set accuracies and G-MIA scores remain much closer to the original model than to RE. Conversely, the layer-wise extensions of GA, FYE, PGU, and SCRUB collapse to random-guess accuracy, while layer-wise BT and ReGUn retain only severely degraded utility. These results expose two distinct failure modes of directly

Table 3: FF unlearning results on VGG-13/CIFAR-10. Complete results on other benchmarks are in Appendix §A. We mark the ineffective removal, impractical efficiency, and model collapse in red.

Method	**Efficiency**	**Effectiveness**		**Utility**
	$\mathrm{t}_{\mathrm{total}}$ (s) ↓	Acc_f(%)→ RE	G-MIA → 0.5	Acc_t(%)↑
Original Model θ_o	$1388.7_{\pm 2.0\%}$	$91.18_{\pm 0.5\%}$	$0.6317_{\pm 0.7\%}$	$82.17_{\pm 0.7\%}$
RE	$1107.0_{\pm 2.1\%}$	$79.26_{\pm 1.1\%}$	$0.5020_{\pm 0.9\%}$	$80.85_{\pm 0.8\%}$
GA	$298.8_{\pm 3.8\%}$	$88.70_{\pm 3.7‰}$	$0.6287_{\pm 3.3\%}$	$80.52_{\pm 1.2\%}$
FYE	$372.0_{\pm 3.2\%}$	$87.89_{\pm 4.2‰}$	$0.6204_{\pm 2.9\%}$	$80.56_{\pm 1.0\%}$
PGU	$315.4_{\pm 3.7\%}$	$88.25_{\pm 4.0‰}$	$0.6277_{\pm 2.7\%}$	$80.89_{\pm 1.2\%}$
SCRUB	$416.1_{\pm 2.7\%}$	$88.84_{\pm 3.1‰}$	$0.6219_{\pm 2.1\%}$	$80.76_{\pm 0.8\%}$
BT	$455.3_{\pm 2.5\%}$	$88.82_{\pm 4.5‰}$	$0.6303_{\pm 3.3\%}$	$80.11_{\pm 1.1\%}$
ReGUn	$453.1_{\pm 2.9\%}$	$88.91_{\pm 4.0‰}$	$0.6351_{\pm 3.2\%}$	$80.23_{\pm 1.2\%}$
(layer-wise) GA	$35.8_{\pm 9.2\%}$	$10.00_{\pm 0.0}$	$0.5027_{\pm 0.3\%}$	$10.00_{\pm 0.0}$
(layer-wise) FYE	$87.2_{\pm 7.6\%}$	$10.00_{\pm 0.0}$	$0.5015_{\pm 0.8\%}$	$10.00_{\pm 0.0}$
(layer-wise) PGU	$69.7_{\pm 7.3\%}$	$10.00_{\pm 0.0}$	$0.5043_{\pm 0.7\%}$	$10.00_{\pm 0.0}$
(layer-wise) SCRUB	$98.8_{\pm 7.5\%}$	$10.00_{\pm 0.0}$	$0.5031_{\pm 0.7\%}$	$10.00_{\pm 0.0}$
(layer-wise) BT	$109.3_{\pm 7.0\%}$	$24.98_{\pm 2.9\%}$	$0.5210_{\pm 1.5\%}$	$27.33_{\pm 1.1\%}$
(layer-wise) ReGUn	$114.8_{\pm 6.5\%}$	$38.12_{\pm 2.7\%}$	$0.5312_{\pm 1.3\%}$	$41.08_{\pm 1.2\%}$
SISA	$1091.8_{\pm 0.7\%}$	$74.91_{\pm 0.9\%}$	$0.5035_{\pm 0.9\%}$	$77.42_{\pm 0.8\%}$
FF-Erase	$139.7_{\pm 3.6\%}$	$81.95_{\pm 1.0\%}$	$0.5135_{\pm 0.8\%}$	$79.48_{\pm 0.7\%}$

transferring BP-oriented objectives to FF models: direct adaptation under-forgets, whereas layer-wise adaptation causes model collapse. Learning-rate sweeps for layer-wise baselines confirm that their failure is not attributable to a single step-size choice (Appendix §C.4).

6.2 Broader Evaluation

We evaluate FF-Erase beyond the standard VGG-13/CIFAR-10 IID setting to demonstrate its broad applicability, all showing the superior performance of FF-Erase. **Across architectures and datasets** (FF-MLP/MNIST to ResNet-50/CIFAR-100), FF-Erase effectively unlearns data and preserves utility, achieving a $3.5\times$-$11.2\times$ speedup over retraining from scratch (RE) (Appendix §A). FF-Erase remains stable across **five real-world unlearning scenarios** (Appendix §E): class-wise, backdoor, mislabeled, non-IID, and high-influence data removal. In these settings, standard BP-based methods often under-forget, and their layer-wise extensions frequently collapse, while FF-Erase consistently realizes effective and efficient unlearning. Furthermore, we demonstrate the **scalability of FF-Erase on forgetting ratios** (Appendix §F). Across 5%–50% deletion ratios, FF-Erase delivers a $5.6\times$–$9.2\times$ speedup over RE, maintaining high unlearning efficacy and utility without ratio-specific tuning. Under **six sequential 5% unlearning requests** (Appendix §G), FF-Erase reuses a single guidance model and remains stable up to 30% cumulative deletion, performing comparably to one-shot runs. Finally, our **layer-wise CKA** analysis (Appendix §H) confirms retraining-like representation changes in the FF hierarchy; 11 of 13 layers remain within 20% of the RE reference. This representation-level evidence complements the prediction and MIA evaluations, confirming the effectiveness of FF-Erase.

6.3 Ablation Study on FF-Erase

We conduct a 4-stage ablation study to empirically support the two FF unlearning challenges introduced in §1, as well as validate the effects of FF-Erase design. This experiment uses the VGG-13 model on the CIFAR-10 dataset and tests 20% IID deletion. We additionally propose Goodness-Shift Consistency (GSC) to quantify the direction differences of unlearning updates between any two layers. Specifically, we define the goodness shift of layer l as $\Delta \boldsymbol{g}^l = \mathrm{softmax}\left(\boldsymbol{g}_u^l\right) - \mathrm{softmax}\left(\boldsymbol{g}_o^l\right)$. Given two layers, the unlearning direction difference is their cosine similarity $\mathrm{CosSim}(\Delta \boldsymbol{g}^{l_1}, \Delta \boldsymbol{g}^{l_2})$. Then GSC is the mean pairwise cosine similarity for a model. A very low GSC indicates divergent layer-wise shifts and is not desired. From Stage I to Stage IV, we progressively augment layer-wise GA with anchor unlearning, add a guidance model for anchor generation, and employ layer-wise λ^l.

Table 4: Ablation study results for FF-Erase. We divide total unlearning time into guidance model obtaining time t_0 and unlearning update time t_1, *i.e.*, t_{total}=t_0+t_1. Model collapse is marked in red.

Method	Stage	GSC	Efficiency			Effectiveness		Utility
			t_0 (s)	t_1 (s)	t_{total} (s)	Acc_f(%)	G-MIA	Acc_t(%)
Layer-wise GA	I	$0.2826_{\pm 4.9\%}$	-	$35.8_{\pm 9.2\%}$	$35.8_{\pm 9.2\%}$	$10.00_{\pm 0.0\%}$	$0.5027_{\pm 0.3\%}$	$10.00_{\pm 0.0\%}$
Mislabeled Anchor	II	$0.5794_{\pm 3.2\%}$	-	$58.4_{\pm 6.8\%}$	$58.4_{\pm 6.8\%}$	$10.00_{\pm 0.0\%}$	$0.5033_{\pm 0.3\%}$	$10.00_{\pm 0.0\%}$
Negative Anchor		$0.5811_{\pm 3.3\%}$	-	$61.2_{\pm 1.1\%}$	$61.2_{\pm 1.1\%}$	$10.00_{\pm 0.0\%}$	$0.5031_{\pm 0.3\%}$	$10.00_{\pm 0.0\%}$
Random Anchor		$0.5615_{\pm 2.7\%}$	-	$55.7_{\pm 7.0\%}$	$55.7_{\pm 7.0\%}$	$10.00_{\pm 0.0\%}$	$0.5048_{\pm 0.3\%}$	$10.00_{\pm 0.0\%}$
Average Anchor		$0.6179_{\pm 2.5\%}$	-	$99.4_{\pm 0.2\%}$	$99.4_{\pm 0.2\%}$	$25.02_{\pm 2.8\%}$	$0.5242_{\pm 1.5\%}$	$26.90_{\pm 2.9\%}$
Weakened Anchor		$0.6784_{\pm 2.3\%}$	-	$87.0_{\pm 0.3\%}$	$87.0_{\pm 0.3\%}$	$33.18_{\pm 2.1\%}$	$0.5305_{\pm 1.3\%}$	$39.09_{\pm 2.0\%}$
Val-Train	III	$0.8387_{\pm 1.3\%}$	$57.1_{\pm 2.3\%}$	$228.4_{\pm 2.9\%}$	$285.5_{\pm 3.6\%}$	$83.12_{\pm 1.4\%}$	$0.5295_{\pm 2.1\%}$	$78.67_{\pm 0.4\%}$
Random-Guidance		$0.6184_{\pm 2.5\%}$	-	$99.2_{\pm 0.2\%}$	$99.2_{\pm 0.2\%}$	$24.32_{\pm 2.9\%}$	$0.5214_{\pm 1.7\%}$	$27.18_{\pm 0.9\%}$
Full-Retrain		$0.8273_{\pm 1.1\%}$	$1107.0_{\pm 2.1\%}$	$72.0_{\pm 2.7\%}$	$1179.0_{\pm 3.4\%}$	$82.08_{\pm 1.3\%}$	$0.5151_{\pm 1.1\%}$	$79.22_{\pm 0.8\%}$
Cross-Domain		$0.8148_{\pm 1.7\%}$	$953.7_{\pm 3.4\%}$	$184.7_{\pm 2.3\%}$	$1138.4_{\pm 3.7\%}$	$81.53_{\pm 1.6\%}$	$0.5138_{\pm 1.7\%}$	$79.42_{\pm 0.9\%}$
FF-Erase (Val-Train)	IV	$0.8287_{\pm 1.2\%}$	$57.1_{\pm 2.3\%}$	$82.6_{\pm 2.9\%}$	$139.7_{\pm 3.1\%}$	$81.95_{\pm 1.0\%}$	$0.5135_{\pm 0.8\%}$	$79.48_{\pm 0.7\%}$

Stage I: Unconstrained layer-wise removal produces divergent directions. Applying gradient ascent independently for every FF layer causes rapid model collapse, with both Acc_f and Acc_t reaching the 10% random-guess level after 35.8 s. Its low GSC of 0.2826 further indicates strongly divergent goodness shifts across layers. This directly supports Challenge (i): independently specifying only what each layer should move away from does not provide coordinated layer-wise destinations.

Stage II: A shared destination improves coordination but remains insufficient. We next force all layers toward the same fixed goodness anchor. Intuitively, we use a mislabeled anchor (the one-hot vector of a wrong label), a negative anchor ($\mathbf{1}$ minus the one-hot vector of y), a random anchor, an average anchor (softmax($\mathbf{1}$)) and a weakened anchor (softmax($\boldsymbol{g}^L/2$)). We observe that variants increase GSC substantially relative to Stage I and delay collapse, with the average and weakened anchors retaining 26.90% and 39.09% utility, respectively. Nevertheless, none reaches

a stable unlearning endpoint. Thus, reducing directional divergence is beneficial but insufficient: heterogeneous FF layers should not be forced toward an identical artificial profile.

Stage III: Empirically valid layer-wise anchors establish stable unlearning directions. We replace the shared anchor with corresponding-layer goodness outputs generated by guidance models. Val-Train, Full-Retrain, and Cross-Domain guidance all avoid collapse and obtain GSC around 0.81–0.84, whereas Random-Guidance remains unstable and causes model collapse. These results indicate that trained FF models expose structured, jointly compatible layer-wise goodness profiles that can serve as stable unlearning anchors, even when the guidance distribution differs from the original training distribution. Additionally, this stage also provides an ablation study on the choice of guidance models. Among feasible guidance models, Val-Train requires the least time (only 57.1 s), approximately 5.2% of full RE time. It is about 16.69× faster than Cross-Domain and 19.38× faster than Full-Retrain, with only a moderate tradeoff in effectiveness (0.0144 G-MIA score increases) and utility (a 0.55% decrease in $\mathrm{Acc_t}$). We therefore adopt it as the default guidance strategy.

Stage IV: Layer-adaptive balancing mitigates heterogeneous gradient-scale imbalance. In this stage, we further enable layer-wise λ^l on the Val-Train guidance models, which forms the complete framework of FF-Erase. In stage III, we observed that even with valid anchors, a globally shared utility weight leaves different layers in different local optimization regimes: some remain removal-dominated while others become utility-dominated (by directly comparing the values of $\|\nabla_{\theta_u^l}\mathcal{L}^l_{\mathrm{removal}}\|_2$ and $\|\nabla_{\theta_u^l}\mathcal{L}^l_{\mathrm{utility}}\|_2$). It is infeasible to set a global λ to balance the two loss objectives for all layers. This empirically supports Challenge (ii). This imbalance disappears in Stage IV, *i.e.*, this layer-adaptive design improves the under/over-forgetting. Moreover, it further reduces the FF updating time substantially (from 228.4s to 82.6s). Meanwhile, GSC remains close to the stable Stage III regime, showing that Stage IV primarily resolves heterogeneous optimization magnitudes rather than changing the established layer-wise directions.

Validation Split. The raw benchmark training set is usually split into a target-training set $\mathbb{D}_{\mathrm{train}}$ and a disjoint validation set $\mathbb{D}_{\mathrm{v}}$. Val-Train strategy just leverages this validation set for a fast guidance model. In Table 5, we keep the same target model θ_o trained on 90% of raw data and only vary the validation ratio to 1%, 3%, 5%, 7%, and 10%. FF-Erase remains stable across a broad range of split ratios. With ratios of 3%–10%, $\mathrm{Acc_f}$ varies by only 0.58 percentage points, G-MIA by 0.0034, and $\mathrm{Acc_t}$ by 0.17 percentage. While the 1% increases $\mathrm{t_{total}}$ to 294.8 s, it still avoids model collapse and retains competitive removal and utility. Therefore, the Val-Train strategy is reliable for obtaining θ_g.

Table 5: Unlearning performance of FF-Erase under different validation split ratios.

Split ratio	GSC	Efficiency			Effectiveness		Utility
		t_0 (s)	t_1 (s)	t_{total} (s)	$\mathrm{Acc_f}$(%)	G-MIA	$\mathrm{Acc_t}$(%)
1%	$0.8091_{\pm 1.3\%}$	$12.2_{\pm 4.1\%}$	$282.6_{\pm 2.4\%}$	$294.8_{\pm 4.5\%}$	$82.98_{\pm 1.1\%}$	$0.5188_{\pm 0.7\%}$	$79.02_{\pm 0.7\%}$
3%	$0.8175_{\pm 1.4\%}$	$30.1_{\pm 3.7\%}$	$160.6_{\pm 2.1\%}$	$190.7_{\pm 4.1\%}$	$82.53_{\pm 1.3\%}$	$0.5106_{\pm 0.7\%}$	$79.31_{\pm 0.6\%}$
5%	$0.8244_{\pm 1.2\%}$	$34.5_{\pm 3.5\%}$	$112.9_{\pm 2.6\%}$	$147.4_{\pm 3.9\%}$	$82.02_{\pm 1.3\%}$	$0.5140_{\pm 0.7\%}$	$79.33_{\pm 0.9\%}$
7%	$0.8233_{\pm 1.1\%}$	$41.2_{\pm 3.0\%}$	$103.7_{\pm 2.4\%}$	$144.9_{\pm 3.3\%}$	$82.03_{\pm 1.1\%}$	$0.5132_{\pm 0.7\%}$	$79.31_{\pm 0.8\%}$
10%	$0.8287_{\pm 1.2\%}$	$57.1_{\pm 2.3\%}$	$82.6_{\pm 2.9\%}$	$139.7_{\pm 3.1\%}$	$81.95_{\pm 1.0\%}$	$0.5135_{\pm 0.8\%}$	$79.48_{\pm 0.7\%}$

7 Conclusion

In this paper, we present the first study of machine unlearning for FF models and identify two fundamental challenges arising from independent layer-wise optimization: different FF layers require coordinated yet layer-specific goodness destinations, while their heterogeneous local dynamics require different under/over-forgetting balancing. Based on these findings, we propose FF-Erase, which uses an efficiently trained guidance model to provide jointly realizable layer-wise goodness anchors and performs layer-wise gradient-magnitude balancing during goodness realignment. This design avoids both the residual knowledge left by final-layer-only adaptation and the model collapse caused by naive layer-wise extensions of BP-oriented methods. We further propose G-MIA, which exploits compact layer-wise goodness information for efficient unlearning verification. Extensive experiments across FF architectures, datasets, forgetting ratios, and practical deletion requests demonstrate that FF-Erase consistently approaches retraining behavior while preserving retained-data utility; layer-wise CKA further confirms retraining-like representation changes throughout the FF hierarchy. Overall, FF-Erase achieves up to 11.2× speedup over retraining and establishes a practical foundation for effective and efficient FF machine unlearning.

AI USE STATEMENT

In this work, we used generative AI tools to polish academic writing, check for potentially missing citations, and confirm the implementation code. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

FF-Erase is intended to support authorized data deletion and model maintenance. G-MIA is introduced as an auditing mechanism, but access to layer-wise goodness may also expose membership information if provided to untrusted parties. We therefore recommend restricting such interfaces to authorized training/tuning/unlearning auditors. Our experiments use standard public benchmark datasets and do not introduce new human-subject data.

REPRODUCIBILITY STATEMENT

We provide the complete FF-Erase and G-MIA objectives in Sections 4 and 5, together with experimental protocols, evaluation metrics, and implementation settings in Section 6 and the appendix. The appendix additionally reports complete benchmark results, practical deletion protocols, forgetting-ratio scalability, and layer-wise CKA analysis. All experiments are repeated over five random seeds. Code and scripts required to reproduce the experiments will be published on acceptance.

REFERENCES

Hassan Ali, Adnan Qayyum, Ala Al-Fuqaha, and Junaid Qadir. Membership inference attacks on dnns using adversarial perturbations. *arXiv preprint arXiv:2307.05193*, 2023.

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

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *2022 IEEE symposium on security and privacy (SP)*, pp. 1897–1914. IEEE, 2022.

Jiyu Chen, Yiwen Guo, Hao Chen, and Neil Gong. Membership inference attack in face of data transformations. In *2022 IEEE Conference on Communications and Network Security (CNS)*, pp. 299–307. IEEE, 2022.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 794–803. PMLR, 10–15 Jul 2018. URL `https://proceedings.mlr.press/v80/chen18a.html`.

Vikram S Chundawat, Ayush K Tarun, Murari Mandal, and Mohan Kankanhalli. Can bad teaching induce forgetting? unlearning in deep networks using an incompetent teacher. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 7210–7217, 2023.

Gilad Cohen and Raja Giryes. Membership inference attack using self influence functions. In *2024 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 4880–4889. IEEE, 2024.

Jiacheng Du, Zhibo Wang, Jie Zhang, Xiaoyi Pang, Jiahui Hu, and Kui Ren. Textual unlearning gives a false sense of unlearning. *arXiv preprint arXiv:2406.13348*, 2024.

Xiangshan Gao, Xingjun Ma, Jingyi Wang, Youcheng Sun, Bo Li, Shouling Ji, Peng Cheng, and Jiming Chen. Verifi: Towards verifiable federated unlearning. *IEEE Transactions on Dependable and Secure Computing*, 21(6):5720–5736, 2024.

Suraj R Gautham, Swapnil Nair, Suresh Jamadagni, Mridul Khurana, and Md Assadi. Exploring the feasibility of forward forward algorithm in neural networks. In *2024 International Conference on Advances in Modern Age Technologies for Health and Engineering Science (AMATHE)*, pp. 1–6. IEEE, 2024.

Yuval Golbari, Navve Wasserman, Gal Vardi, and Michal Irani. Impmia: Leveraging implicit bias for membership inference attack. *arXiv preprint arXiv:2510.10625*, 2025.

Elizabeth Liz Harding, Jarno J Vanto, Reece Clark, L Hannah Ji, and Sara C Ainsworth. Understanding the scope and impact of the california consumer privacy act of 2018. *Journal of Data Protection & Privacy*, 2(3):234–253, 2019.

Yun He, Xue Feng, Cheng Cheng, Geng Ji, Yunsong Guo, and James Caverlee. Metabalance: Improving multi-task recommendations via adapting gradient magnitudes of auxiliary tasks. In *Proceedings of the ACM Web Conference 2022*, WWW '22, pp. 2205–2215, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450390965. doi: 10.1145/3485447.3512093. URL https://doi.org/10.1145/3485447.3512093.

Geoffrey Hinton. The forward-forward algorithm: Some preliminary investigations, 2022.

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

Bargav Jayaraman, Lingxiao Wang, Katherine Knipmeyer, Quanquan Gu, and David Evans. Revisiting membership inference under realistic assumptions. *arXiv preprint arXiv:2005.10881*, 2020.

Yongwoo Kim, Sungmin Cha, and Donghyun Kim. Are we truly forgetting? a critical re-examination of machine unlearning evaluation protocols. *arXiv preprint arXiv:2503.06991*, 2025.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International conference on machine learning*, pp. 3519–3529. PMlR, 2019.

Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. Towards unbounded machine unlearning. *Advances in neural information processing systems*, 36:1957–1987, 2023.

Heung-Chang Lee and Jeonggeun Song. Symba: Symmetric backpropagation-free contrastive learning with forward-forward algorithm for optimizing convergence, 2023.

Klas Leino and Matt Fredrikson. Stolen memories: Leveraging model memorization for calibrated {White-Box} membership inference. In *29th USENIX security symposium (USENIX Security 20)*, pp. 1605–1622, 2020.

Lan Liu, Yi Wang, Gaoyang Liu, Kai Peng, and Chen Wang. Membership inference attacks against machine learning models via prediction sensitivity. *IEEE Transactions on Dependable and Secure Computing*, 20(3):2341–2347, 2022a.

Yi Liu, Lei Xu, Xingliang Yuan, Cong Wang, and Bo Li. The right to be forgotten in federated learning: An efficient realization with rapid retraining. In *IEEE INFOCOM 2022-IEEE conference on computer communications*, pp. 1749–1758. IEEE, 2022b.

Jonas Mirlach, Sonia Laguna, and Julia E. Vogt. Reference-guided machine unlearning, 2026. URL https://arxiv.org/abs/2603.11210.

Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE symposium on security and privacy (SP)*, pp. 739–753. IEEE, 2019.

Andreas Papachristodoulou, Christos Kyrkou, Stelios Timotheou, and Theocharis Theocharides. Convolutional channel-wise competitive learning for the forward-forward algorithm. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 14536–14544, 2024.

Namyong Park, Xing Wang, Antoine Simoulin, Shuai Yang, Grey Yang, Ryan Rossi, Puja Trivedi, and Nesreen Ahmed. Forward learning of graph neural networks. *arXiv preprint arXiv:2403.11004*, 2024.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. *Advances in neural information processing systems*, 33: 19920–19930, 2020.

Xinbao Qiao, Meng Zhang, Ming Tang, and Ermin Wei. Hessian-free online certified unlearning. *arXiv preprint arXiv:2404.01712*, 2024.

Deepak Ravikumar, Efstathia Soufleri, and Kaushik Roy. Curvature clues: Decoding deep learning privacy with input loss curvature. *Advances in Neural Information Processing Systems*, 37: 20003–20030, 2024.

Ayush Sekhari, Jayadev Acharya, Gautam Kamath, and Ananda Theertha Suresh. Remember what you want to forget: Algorithms for machine unlearning. *Advances in Neural Information Processing Systems*, 34:18075–18086, 2021.

Liang Sun, Yang Zhang, Weizhao He, Jiajun Wen, Linlin Shen, and Weicheng Xie. Deeperforward: Enhanced forward-forward training for deeper and better performance. In *The Thirteenth International Conference on Learning Representations*, 2025.

Anshuman Suri, Xiao Zhang, and David Evans. Do parameters reveal more than loss for membership inference? *arXiv preprint arXiv:2406.11544*, 2024.

Ayush K Tarun, Vikram S Chundawat, Murari Mandal, and Mohan Kankanhalli. Fast yet effective machine unlearning. *IEEE Transactions on Neural Networks and Learning Systems*, 35(9):13046–13055, 2023.

Mauricio Ortiz Torres, Markus Lange, and Arne P Raulf. On advancements of the forward-forward algorithm. In *2025 11th International Conference on Control, Decision and Information Technologies (CoDIT)*, volume 1, pp. 2075–2080. IEEE, 2025.

Paul Voigt and Axel Von dem Bussche. The eu general data protection regulation (gdpr). *A practical guide, 1st ed., Cham: Springer International Publishing*, 10(3152676):10–5555, 2017.

Tinghua Wang, Dongyan Zhao, and Shengfeng Tian. An overview of kernel alignment and its applications. *Artificial Intelligence Review*, 43(2):179–192, 2015.

Wenbin Wang, Qiwen Ma, Zifan Zhang, Yuchen Liu, Zhuqing Liu, and Minghong Fang. Poisoning attacks and defenses to federated unlearning. In *Companion Proceedings of the ACM on Web Conference 2025*, pp. 1365–1369, 2025.

Siyuan Xu, Yujie Wu, Jibin Wu, Lei Deng, Mingkun Xu, Qinghao Wen, and Guoqi Li. Advancing the forward-forward algorithm towards high-performance deep local learning. *Neural Networks*, pp. 108765, 2026.

Xiaoyu Xu, Xiang Yue, Yang Liu, Qingqing Ye, Huadi Zheng, Peizhao Hu, Minxin Du, and Haibo Hu. Unlearning isn't deletion: Investigating reversibility of machine unlearning in llms. *arXiv preprint arXiv:2505.16831*, 2025.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014.

Sajjad Zarifzadeh, Philippe Liu, and Reza Shokri. Low-cost high-power membership inference attacks. *arXiv preprint arXiv:2312.03262*, 2023.

Haibo Zhang, Toru Nakamura, Takamasa Isohara, and Kouichi Sakurai. A review on machine unlearning. *SN Computer Science*, 4(4):337, 2023.

A Additional FF Unlearning Experimental Results

We further evaluate FF-Erase across additional datasets and FF architectures under the same 20% IID deletion setting. Tables 6, 7, 8, 9, and 10 cover VGG-13 on CIFAR-100, MNIST, and Fashion-MNIST, ResNet-50 on CIFAR-100, and the original FF-MLP setting on MNIST, respectively. Together with the VGG-13/CIFAR-10 results in the main text, these experiments evaluate FF-Erase across different data complexities, class counts, network depths, and FF architectures.

FF-Erase consistently approaches retraining behavior across benchmarks. Across all five additional settings, FF-Erase achieves a $3.5\times$ to $11.2\times$ speedup over RE. More importantly, its forgetting-set accuracy remains within 1.70 percentage points of RE in every case, while the G-MIA gap from RE is at most 0.0103. Retained-data utility is similarly stable, with the test-accuracy gap from RE remaining below 1.0 percentage point across all settings. The same behavior holds from the original FF-MLP on MNIST to the substantially deeper ResNet-50 on CIFAR-100, demonstrating that FF-Erase is not tied to a particular FF architecture or dataset scale.

Existing BP-oriented methods exhibit consistent failure modes across settings. Direct adaptations of GA, FYE, PGU, SCRUB, BT, and ReGUn generally preserve test utility but leave substantial residual membership information, with G-MIA scores remaining much closer to the original model than to RE. This confirms that modifying only the final FF layer does not remove the knowledge retained by intermediate layers. Conversely, applying the same objectives independently to every FF layer frequently causes severe utility degradation or model collapse. Thus, simply extending a BP-oriented objective across FF layers does not resolve the layer-wise unlearning problem.

Retraining-based alternatives remove information but sacrifice efficiency. SISA generally obtains G-MIA scores close to RE, but requires 94%–105% of the full retraining time and often incurs a larger utility gap. In contrast, FF-Erase combines retraining-like removal and utility with substantially lower computational cost. These results consistently support the central design of FF-Erase: coordinated layer-specific goodness targets and layer-adaptive optimization are necessary to achieve effective and efficient unlearning across heterogeneous FF models.

Table 6: FF unlearning results on VGG-13/CIFAR-100. We mark the ineffective removal, impractical efficiency and model collapse in red.

Method	**Efficiency**	**Effectiveness**		**Utility**
	t_{total} (s) $\downarrow$	$Acc_f(\%) \rightarrow$ RE	G-MIA $\rightarrow 0.5$	$Acc_t(\%)\uparrow$
Original Model θ_o	$1924.53_{\pm 2.1\%}$	$61.57_{\pm 0.6\%}$	$0.6340_{\pm 0.8\%}$	$53.43_{\pm 0.8\%}$
RE	$1444.86_{\pm 2.2\%}$	$54.00_{\pm 1.2\%}$	$0.5040_{\pm 1.0\%}$	$52.63_{\pm 0.9\%}$
GA	$523.99_{\pm 3.9\%}$	$61.00_{\pm 3.8‰}$	$0.6312_{\pm 3.4\%}$	$52.43_{\pm 1.3\%}$
FYE	$417.53_{\pm 3.3\%}$	$59.48_{\pm 4.3‰}$	$0.6229_{\pm 3.0\%}$	$52.46_{\pm 1.1\%}$
PGU	$548.66_{\pm 3.8\%}$	$59.71_{\pm 4.1‰}$	$0.6302_{\pm 2.8\%}$	$52.65_{\pm 1.3\%}$
SCRUB	$543.09_{\pm 2.8\%}$	$60.08_{\pm 3.2‰}$	$0.6244_{\pm 2.2\%}$	$52.58_{\pm 0.9\%}$
BT	$594.26_{\pm 2.6\%}$	$60.07_{\pm 4.6‰}$	$0.6328_{\pm 3.4\%}$	$52.18_{\pm 1.2\%}$
ReGUn	$631.39_{\pm 3.0\%}$	$60.13_{\pm 4.1‰}$	$0.6346_{\pm 3.3\%}$	$52.26_{\pm 1.3\%}$
(layer-wise) GA	$66.73_{\pm 9.3\%}$	$1.00_{\pm 0.0}$	$0.5007_{\pm 0.4\%}$	$1.00_{\pm 0.0}$
(layer-wise) FYE	$113.81_{\pm 7.7\%}$	$1.00_{\pm 0.0}$	$0.5095_{\pm 0.9\%}$	$1.00_{\pm 0.0}$
(layer-wise) PGU	$70.97_{\pm 7.4\%}$	$1.00_{\pm 0.0}$	$0.5023_{\pm 0.8\%}$	$1.00_{\pm 0.0}$
(layer-wise) SCRUB	$118.95_{\pm 7.6\%}$	$1.00_{\pm 0.0}$	$0.5011_{\pm 0.8\%}$	$1.00_{\pm 0.0}$
(layer-wise) BT	$162.66_{\pm 7.1\%}$	$7.49_{\pm 3.0\%}$	$0.5231_{\pm 1.6\%}$	$11.64_{\pm 1.2\%}$
(layer-wise) ReGUn	$169.84_{\pm 6.6\%}$	$18.05_{\pm 2.8\%}$	$0.5333_{\pm 1.4\%}$	$16.33_{\pm 1.3\%}$
SISA	$1425.02_{\pm 0.8\%}$	$51.23_{\pm 1.0\%}$	$0.5055_{\pm 1.0\%}$	$50.56_{\pm 0.9\%}$
FF-Erase	$186.34_{\pm 3.7\%}$	$55.70_{\pm 1.1\%}$	$0.5126_{\pm 0.9\%}$	$51.80_{\pm 0.8\%}$

B Alternative Guidance-Model Construction Strategies

The default Val-Train strategy constructs the guidance model from a held-out validation split. To examine whether FF-Erase is intrinsically dependent on such a split, we further plug three alternative forms of guidance into the complete layer-adaptive FF-Erase pipeline. For a fraction ρ, *Mini-Retrain* trains a fresh FF guidance model from scratch on a randomly sampled ρ fraction of the retained set $\mathbb{D}_r$, while *Fast-Distill* trains a fresh guidance model through distillation on the same retained-data budget. *Cross-Domain* trains the guidance model on a ten-class subset of CIFAR-100, whose class count matches the CIFAR-10 FF output space but whose input distribution differs from the target-training data. All guidance models use the same architecture as the target model and remain frozen during unlearning.

Table 7: FF unlearning results on VGG-13/MNIST. We mark the ineffective removal, impractical efficiency and model collapse in red.

Method	Efficiency	Effectiveness		Utility
	t_{total} (s) ↓	Acc_f(%)→ RE	G-MIA → 0.5	Acc_t(%)↑
Original Model θ_o	$436.55_{\pm 2.2\%}$	$99.98_{\pm 0.0\%}$	$0.6166_{\pm 0.9\%}$	$99.50_{\pm 0.2\%}$
RE	$362.21_{\pm 1.9\%}$	$99.85_{\pm 0.1\%}$	$0.5017_{\pm 0.7\%}$	$99.41_{\pm 0.2\%}$
GA	$119.64_{\pm 4.0\%}$	$95.42_{\pm 3.9‰}$	$0.6089_{\pm 3.5\%}$	$98.64_{\pm 1.4\%}$
FYE	$122.75_{\pm 3.1\%}$	$96.14_{\pm 4.1‰}$	$0.5892_{\pm 2.8\%}$	$99.31_{\pm 0.9\%}$
PGU	$122.38_{\pm 3.7\%}$	$97.31_{\pm 4.0‰}$	$0.6015_{\pm 2.7\%}$	$99.09_{\pm 1.2\%}$
SCRUB	$138.16_{\pm 2.8\%}$	$96.82_{\pm 3.2‰}$	$0.5919_{\pm 2.2\%}$	$99.16_{\pm 0.9\%}$
BT	$143.45_{\pm 2.4\%}$	$97.48_{\pm 4.4‰}$	$0.6085_{\pm 3.2\%}$	$98.04_{\pm 1.0\%}$
ReGUn	$141.59_{\pm 3.1\%}$	$97.36_{\pm 4.2‰}$	$0.5988_{\pm 3.4\%}$	$98.99_{\pm 1.4\%}$
(layer-wise) GA	$21.54_{\pm 9.0\%}$	$10.00_{\pm 0.0}$	$0.5042_{\pm 0.6\%}$	$10.00_{\pm 0.0}$
(layer-wise) FYE	$52.11_{\pm 7.7\%}$	$10.00_{\pm 0.0}$	$0.5072_{\pm 0.9\%}$	$10.00_{\pm 0.0}$
(layer-wise) PGU	$36.03_{\pm 7.1\%}$	$10.00_{\pm 0.0}$	$0.5068_{\pm 0.5\%}$	$10.00_{\pm 0.0}$
(layer-wise) SCRUB	$51.98_{\pm 7.7\%}$	$10.00_{\pm 0.0}$	$0.5006_{\pm 0.9\%}$	$10.00_{\pm 0.0}$
(layer-wise) BT	$55.58_{\pm 6.9\%}$	$36.60_{\pm 2.8\%}$	$0.5181_{\pm 1.4\%}$	$37.49_{\pm 1.0\%}$
(layer-wise) ReGUn	$53.74_{\pm 6.5\%}$	$45.09_{\pm 2.7\%}$	$0.5235_{\pm 1.3\%}$	$48.44_{\pm 1.2\%}$
SISA	$356.39_{\pm 0.8\%}$	$99.31_{\pm 1.0\%}$	$0.5021_{\pm 1.0\%}$	$98.79_{\pm 0.9\%}$
FF-Erase	$83.57_{\pm 3.4\%}$	$99.78_{\pm 0.2\%}$	$0.5120_{\pm 0.6\%}$	$99.36_{\pm 0.5\%}$

Table 8: FF unlearning results on VGG-13/Fashion-MNIST. We mark the ineffective removal, impractical efficiency and model collapse in red.

Method	Efficiency	Effectiveness		Utility
	t_{total} (s) ↓	Acc_f(%)→ RE	G-MIA → 0.5	Acc_t(%)↑
Original Model θ_o	$840.73_{\pm 2.3\%}$	$99.48_{\pm 0.3\%}$	$0.8217_{\pm 1.0\%}$	$94.53_{\pm 0.3\%}$
RE	$665.52_{\pm 2.0\%}$	$98.35_{\pm 0.2\%}$	$0.5067_{\pm 0.8\%}$	$93.07_{\pm 0.3\%}$
GA	$180.54_{\pm 4.1\%}$	$98.91_{\pm 4.0‰}$	$0.7839_{\pm 3.6\%}$	$92.13_{\pm 1.5\%}$
FYE	$223.86_{\pm 3.2\%}$	$99.62_{\pm 4.2‰}$	$0.8141_{\pm 2.9\%}$	$92.80_{\pm 1.0\%}$
PGU	$191.33_{\pm 3.8\%}$	$99.28_{\pm 4.1‰}$	$0.8065_{\pm 2.8\%}$	$91.58_{\pm 1.3\%}$
SCRUB	$249.40_{\pm 2.9\%}$	$99.30_{\pm 3.3‰}$	$0.7968_{\pm 2.3\%}$	$91.66_{\pm 1.0\%}$
BT	$274.82_{\pm 2.5\%}$	$98.97_{\pm 4.5‰}$	$0.8166_{\pm 3.3\%}$	$92.53_{\pm 1.1\%}$
ReGUn	$272.95_{\pm 3.2\%}$	$99.15_{\pm 4.3‰}$	$0.7931_{\pm 3.5\%}$	$92.48_{\pm 1.5\%}$
(layer-wise) GA	$21.65_{\pm 9.1\%}$	$10.00_{\pm 0.0}$	$0.5082_{\pm 0.7\%}$	$10.00_{\pm 0.0}$
(layer-wise) FYE	$52.37_{\pm 7.8\%}$	$10.00_{\pm 0.0}$	$0.5052_{\pm 1.0\%}$	$10.00_{\pm 0.0}$
(layer-wise) PGU	$42.24_{\pm 7.2\%}$	$10.00_{\pm 0.0}$	$0.5008_{\pm 0.2\%}$	$10.00_{\pm 0.0}$
(layer-wise) SCRUB	$59.27_{\pm 7.8\%}$	$10.00_{\pm 0.0}$	$0.5036_{\pm 1.0\%}$	$10.00_{\pm 0.0}$
(layer-wise) BT	$65.91_{\pm 7.0\%}$	$36.78_{\pm 2.9\%}$	$0.6654_{\pm 1.5\%}$	$27.68_{\pm 1.1\%}$
(layer-wise) ReGUn	$69.08_{\pm 6.6\%}$	$45.32_{\pm 2.8\%}$	$0.6879_{\pm 1.4\%}$	$38.68_{\pm 1.3\%}$
SISA	$659.67_{\pm 0.9\%}$	$97.81_{\pm 1.1\%}$	$0.5075_{\pm 1.1\%}$	$92.28_{\pm 1.0\%}$
FF-Erase	$113.99_{\pm 3.5\%}$	$98.28_{\pm 0.3\%}$	$0.5129_{\pm 0.7\%}$	$92.86_{\pm 0.6\%}$

Unlike Stage III in Table 4, which uses a globally shared utility weight to isolate the effect of anchor construction, the following experiment inserts every guidance strategy into the complete layer-adaptive FF-Erase pipeline. We evaluate VGG-13/CIFAR-10 with 20% IID deletion. The guidance-construction time, FF update time, and total cost are denoted by t_0, t_1, and $t_{\text{total}} = t_0 + t_1$, respectively.

Results. Table 11 shows that the guidance-driven realignment mechanism is not tied to a held-out validation set. Mini-Retrain and Fast-Distill both avoid model collapse and yield substantially more retraining-like behavior than the BP-oriented baselines, even though their guidance models are constructed only from subsets of $\mathbb{D}_r$. Fast-Distill with $\rho = 0.30$ and Cross-Domain guidance obtain the smallest G-MIA gaps to RE, demonstrating that useful layer-wise anchors can be learned from either retained or distribution-shifted data.

Val-Train nevertheless provides the best end-to-end efficiency: its total cost is 139.7 s, compared with 199.9–272.8 s for Mini-Retrain, 207.2–362.4 s for Fast-Distill, and 1015.5 s for Cross-Domain guidance. It also requires no access to $\mathbb{D}_r$ for guidance construction, introduces no request-specific guidance-data fraction ρ, and can be reused across sequential deletion requests. We therefore use Val-Train as the default strategy, while the alternative results establish that FF-Erase itself does not fundamentally depend on validation-set guidance.

Table 9: FF unlearning results on ResNet-50/CIFAR-100. We mark the ineffective removal, impractical efficiency and model collapse in red.

Method	Efficiency	Effectiveness		Utility
	t_{total} (s) ↓	Acc_f(%)→ RE	G-MIA → 0.5	Acc_t(%)↑
Original Model θ_o	$19396.5_{\pm 1.7\%}$	$66.39_{\pm 0.7\%}$	$0.7755_{\pm 0.9\%}$	$59.18_{\pm 0.9\%}$
RE	$16225.3_{\pm 2.1\%}$	$58.11_{\pm 1.3\%}$	$0.5093_{\pm 1.1\%}$	$57.48_{\pm 1.0\%}$
GA	$2173.4_{\pm 4.0\%}$	$61.76_{\pm 3.9‰}$	$0.7527_{\pm 3.5\%}$	$54.26_{\pm 1.4\%}$
FYE	$3195.8_{\pm 3.4\%}$	$62.10_{\pm 4.4‰}$	$0.7444_{\pm 3.1\%}$	$55.28_{\pm 1.2\%}$
PGU	$2405.4_{\pm 3.9\%}$	$61.35_{\pm 4.2‰}$	$0.7517_{\pm 2.9\%}$	$53.50_{\pm 1.4\%}$
SCRUB	$2811.8_{\pm 2.9\%}$	$61.77_{\pm 3.3‰}$	$0.7459_{\pm 2.3\%}$	$53.42_{\pm 1.0\%}$
BT	$3359.4_{\pm 2.7\%}$	$61.76_{\pm 4.7‰}$	$0.7543_{\pm 3.5\%}$	$54.96_{\pm 1.3\%}$
ReGUn	$3328.6_{\pm 3.1\%}$	$61.82_{\pm 4.2‰}$	$0.7561_{\pm 3.4\%}$	$54.05_{\pm 1.4\%}$
(layer-wise) GA	$400.1_{\pm 5.4\%}$	$1.00_{\pm 0.0}$	$0.5119_{\pm 0.5\%}$	$1.00_{\pm 0.0}$
(layer-wise) FYE	$617.9_{\pm 4.8\%}$	$1.00_{\pm 0.0}$	$0.5098_{\pm 1.0\%}$	$1.00_{\pm 0.0}$
(layer-wise) PGU	$773.5_{\pm 4.5\%}$	$1.00_{\pm 0.0}$	$0.5135_{\pm 0.9\%}$	$1.00_{\pm 0.0}$
(layer-wise) SCRUB	$779.9_{\pm 4.7\%}$	$1.00_{\pm 0.0}$	$0.5123_{\pm 0.9\%}$	$1.00_{\pm 0.0}$
(layer-wise) BT	$826.7_{\pm 4.2\%}$	$1.00_{\pm 0.0}$	$0.5144_{\pm 1.7\%}$	$1.00_{\pm 0.0}$
(layer-wise) ReGUn	$813.5_{\pm 4.7\%}$	$1.00_{\pm 0.0}$	$0.5046_{\pm 1.5\%}$	$1.00_{\pm 0.0}$
SISA	$15249.6_{\pm 0.9\%}$	$55.09_{\pm 1.1\%}$	$0.5168_{\pm 1.1\%}$	$54.08_{\pm 1.0\%}$
FF-Erase	$1451.3_{\pm 3.8\%}$	$58.37_{\pm 1.2\%}$	$0.5139_{\pm 1.0\%}$	$56.51_{\pm 0.9\%}$

Table 10: FF unlearning results on MLP/MNIST (original setting in Hinton (2022)). We mark the ineffective removal, impractical efficiency and model collapse in red.

Method	Efficiency	Effectiveness		Utility
	t_{total} (s) ↓	Acc_f(%)→ RE	G-MIA → 0.5	Acc_t(%)↑
Original Model θ_o	$412.7_{\pm 1.8\%}$	$99.89_{\pm 0.1\%}$	$0.6294_{\pm 1.1\%}$	$98.58_{\pm 0.7\%}$
RE	$336.9_{\pm 2.0\%}$	$99.61_{\pm 1.5\%}$	$0.5062_{\pm 0.9\%}$	$97.28_{\pm 1.1\%}$
GA	$79.7_{\pm 4.2\%}$	$98.46_{\pm 3.8‰}$	$0.5963_{\pm 3.6\%}$	$84.09_{\pm 1.2\%}$
FYE	$87.4_{\pm 3.2\%}$	$98.39_{\pm 4.5‰}$	$0.5920_{\pm 3.3\%}$	$85.71_{\pm 1.1\%}$
PGU	$86.6_{\pm 4.0\%}$	$98.15_{\pm 4.0‰}$	$0.6142_{\pm 3.1\%}$	$83.24_{\pm 1.3\%}$
SCRUB	$92.4_{\pm 3.1\%}$	$99.27_{\pm 3.2‰}$	$0.6122_{\pm 2.4\%}$	$83.67_{\pm 0.8\%}$
BT	$97.1_{\pm 2.6\%}$	$98.27_{\pm 4.9‰}$	$0.6055_{\pm 3.3\%}$	$84.79_{\pm 1.4\%}$
ReGUn	$92.5_{\pm 3.3\%}$	$99.42_{\pm 4.1‰}$	$0.5943_{\pm 3.5\%}$	$84.39_{\pm 1.2\%}$
(layer-wise) GA	$15.2_{\pm 5.2\%}$	$10.00_{\pm 0.0}$	$0.5098_{\pm 0.6\%}$	$10.00_{\pm 0.0}$
(layer-wise) FYE	$53.7_{\pm 4.9\%}$	$22.89_{\pm 4.1‰}$	$0.5684_{\pm 3.6\%}$	$19.72_{\pm 4.0‰}$
(layer-wise) PGU	$57.6_{\pm 4.3\%}$	$24.77_{\pm 4.4‰}$	$0.5533_{\pm 3.3\%}$	$22.96_{\pm 4.3‰}$
(layer-wise) SCRUB	$36.8_{\pm 4.8\%}$	$10.00_{\pm 0.0}$	$0.5159_{\pm 0.7\%}$	$10.00_{\pm 0.0}$
(layer-wise) BT	$66.9_{\pm 4.1\%}$	$33.72_{\pm 4.4‰}$	$0.5790_{\pm 3.2\%}$	$30.76_{\pm 4.3‰}$
(layer-wise) ReGUn	$69.4_{\pm 4.9\%}$	$27.93_{\pm 4.0‰}$	$0.5644_{\pm 3.5\%}$	$29.88_{\pm 4.1‰}$
SISA	$353.7_{\pm 1.0\%}$	$99.46_{\pm 0.9\%}$	$0.5083_{\pm 1.3\%}$	$94.79_{\pm 0.9\%}$
FF-Erase	$96.1_{\pm 3.6\%}$	$99.57_{\pm 1.4\%}$	$0.5129_{\pm 0.9\%}$	$97.21_{\pm 1.0\%}$

C Implementation Details

C.1 Implementation on FF-Erase

Optimization Setup. Unless otherwise specified, we use a batch size of 64 and a learning rate of η =1e-4, *i.e.*, $\eta = 1 \times 10^{-4}$, for FF-Erase unlearning. All experiments are implemented in PyTorch and executed on NVIDIA A800 GPUs. The same optimization configuration is used across datasets, architectures, forgetting ratios, and deletion requests without scenario-specific tuning.

Local Early Stopping. We use the same fixed layer-wise early-stopping rule for all experiments. For each layer l, we monitor its complete local FF-Erase objective, $\mathcal{L}^l_{FF-Erase,t} = \mathcal{L}^l_{\text{removal},t} + \lambda^l_t \mathcal{L}^l_{\text{utility},t}$, and maintain its best-so-far value. An improvement at epoch t is considered meaningful when $\frac{\mathcal{L}^l_{FF-Erase,\text{best}} - \mathcal{L}^l_{FF-Erase,t}}{\mathcal{L}^l_{FF-Erase,\text{best}} + \epsilon_{\text{num}}} \geq \epsilon_{\text{es}}$, where ϵ_{num} is used only for numerical stability. If no meaningful improvement is observed for P consecutive epochs, layer l is frozen while the remaining layers continue optimization. We fix $\epsilon_{\text{es}} = 1\%$, $P = 3$, and a maximum of $E_{\max} = 200$ unlearning epochs globally for all experiments.

To avoid prematurely stopping a downstream layer while its input representation is still changing, the stopping criterion is activated sequentially: layer $l + 1$ starts accumulating its early-stopping statistics only after layer l has terminated. This is consistent with the greedy layer-wise nature of FF optimization, where updates to an upstream layer change the input distribution observed by subsequent layers. The early-stopping rule only determines when a locally stabilized layer terminates optimization and does not modify the FF-Erase objective or its layer-adaptive gradient balancing.

Table 11: FF-Erase performance with alternative guidance-model construction strategies on VGG-13/CIFAR-10 with 20% IID deletion. We mark the ineffective removal, impractical efficiency and model collapse in red. The "Dataset" means what this strategy require for obtaining the guidance model.

Strategy	Dataset	Efficiency			Effectiveness		Utility
		t_0 (s) ↓	t_1 (s) ↓	t_{total} (s) ↓	Acc_f(%)→ RE	G-MIA → 0.5	Acc_t(%)↑
RE	CIFAR-10 $\mathbb{D}_r$	-	$1107.0_{\pm 2.1\%}$	$1107.0_{\pm 2.1\%}$	$79.26_{\pm 1.1\%}$	$0.5020_{\pm 0.9\%}$	$80.85_{\pm 0.8\%}$
Val-Train	CIFAR-10 $\mathbb{D}_{valid}$	$57.1_{\pm 2.3\%}$	$82.6_{\pm 2.9\%}$	$139.7_{\pm 3.6\%}$	$81.95_{\pm 1.0\%}$	$0.5135_{\pm 0.8\%}$	$79.48_{\pm 0.7\%}$
Mini-Retrain (0.15)	CIFAR-10 $\mathbb{D}_r$	$72.5_{\pm 3.1\%}$	$127.4_{\pm 2.1\%}$	$199.9_{\pm 4.0\%}$	$82.33_{\pm 1.8\%}$	$0.5343_{\pm 2.1\%}$	$78.92_{\pm 0.9\%}$
Mini-Retrain (0.3)	CIFAR-10 $\mathbb{D}_r$	$197.7_{\pm 4.3\%}$	$75.1_{\pm 2.2\%}$	$272.8_{\pm 4.6\%}$	$82.97_{\pm 1.4\%}$	$0.5311_{\pm 1.7\%}$	$79.24_{\pm 0.7\%}$
Fast-Distill (0.15)	CIFAR-10 $\mathbb{D}_r$	$93.0_{\pm 3.7\%}$	$114.2_{\pm 3.0\%}$	$207.2_{\pm 4.4\%}$	$81.18_{\pm 1.2\%}$	$0.5278_{\pm 1.4\%}$	$79.05_{\pm 0.8\%}$
Fast-Distill (0.3)	CIFAR-10 $\mathbb{D}_r$	$255.9_{\pm 4.7\%}$	$106.5_{\pm 2.7\%}$	$362.4_{\pm 5.1\%}$	$80.53_{\pm 1.1\%}$	$0.5105_{\pm 0.9\%}$	$80.13_{\pm 0.8\%}$
Cross-Domain	CIFAR-100 $\mathbb{D}_{train}$ subset	$953.7_{\pm 3.4\%}$	$61.8_{\pm 1.5\%}$	$1015.5_{\pm 3.9\%}$	$80.12_{\pm 1.3\%}$	$0.5083_{\pm 0.8\%}$	$80.15_{\pm 0.8\%}$

Layer-wise Gradient Balancing. For every layer l, the adaptive utility coefficient is computed as $\lambda_t^l = \mathrm{sg}\left(\frac{\left\|\nabla_{\theta_u^l}\mathcal{L}^l_{\text{removal}}\right\|_2}{\left\|\nabla_{\theta_u^l}\mathcal{L}^l_{\text{utility}}\right\|_2+\epsilon_{\text{num}}}\right)$, where $\mathrm{sg}(\cdot)$ denotes stop-gradient and $\epsilon_{\text{num}} = 10^{-3}$ is a fixed numerical constant. Thus, λ_t^l is treated as a scalar coefficient when the final parameter update is differentiated; no second-order gradient through λ_t^l is introduced. We use no additional clipping or smoothing of λ_t^l in the reported experiments.

The predictor. The predictor is parameter-free and aggregates the layer-wise goodness vectors by weighted averaging, where the aggregating weight is determined by validation set during training. Therefore, only the FF-layer parameters $\{\theta^l\}_{l=1}^L$ are updated during unlearning.

C.2 Implementation on the Validation Set

Before training the original model, we partition the raw benchmark training split into a target-training set $\mathbb{D}_{\text{train}}$ and a held-out validation reservoir $\mathbb{D}_{\text{v}}^{\max}$, where $|\mathbb{D}_{\text{v}}^{\max}|$ equals 10% of the raw training split. This corresponds to 5,000 held-out samples for CIFAR-10/CIFAR-100 and 6,000 samples for MNIST/Fashion-MNIST. The original model is trained only on $\mathbb{D}_{\text{train}}$, and all forgetting requests are subsequently sampled from $\mathbb{D}_{\text{train}}$. Therefore, $\mathbb{D}_{\text{v}}^{\max} \cap \mathbb{D}_{\text{train}} = \varnothing, \qquad \mathbb{D}_{\text{v}}^{\max} \cap \mathbb{D}_{\text{f}} = \varnothing.$

In the experiments, we split 10% of the original raw training samples as the validation set for early stopping in training and Val-Train guidance. This split ratio is fixed globally across datasets and model architectures.

C.3 Implementation on G-MIA

Implementation on G-MIA evaluation: We separate the evaluation of G-MIA as a membership classifier from its use as an unlearning auditor. When reporting the G-MIA performance (Table 1 in §5 and Table 16- 18 in Appendix §D), we divide the test set $\mathbb{D}_{\text{test}}$ into disjoint $\mathbb{D}_{\text{test}}^0$ and $\mathbb{D}_{\text{test}}^1$ with same amount of samples inside, i.e., $\mathbb{D}_{\text{test}} = \mathbb{D}_{\text{test}}^0 \cup \mathbb{D}_{\text{test}}^1$, $\mathbb{D}_{\text{test}}^0 \cap \mathbb{D}_{\text{test}}^1 = \emptyset$ and $|\mathbb{D}_{\text{test}}^0| = |\mathbb{D}_{\text{test}}^1|$. We use $\mathbb{D}_{\text{test}}^0$ as the non-member data and sample same number of training samples $\mathbb{D}_{\text{train}}^0 \subset \mathbb{D}_{\text{train}}$ ($|\mathbb{D}_{\text{train}}^0| = |\mathbb{D}_{\text{test}}^0|$) as the member data to train the G-MIA. Next, we sample $\mathbb{D}_{\text{train}}^1 \subset \mathbb{D}_{\text{train}}$ such that $\mathbb{D}_{\text{train}}^0 \cap \mathbb{D}_{\text{train}}^1 = \emptyset$ and $|\mathbb{D}_{\text{train}}^1| = |\mathbb{D}_{\text{test}}^1|$. We use $|\mathbb{D}_{\text{test}}^1|$ and $\mathbb{D}_{\text{train}}^1$ as the test set for G-MIA to report its ACC and AUC score.

Implementation on G-MIA for unlearning verification: For unlearning verification, we train the attacker on the original model using retained members $\mathbb{D}_{\text{r}}^{\text{atk}} \subset \mathbb{D}_{\text{r}}$ and non-members $\mathbb{D}_{\text{test}}^{\text{atk}} \subset \mathbb{D}_{\text{test}}$. The attacker is then frozen and evaluated on the forgetting set $\mathbb{D}_{\text{f}}$ and a disjoint non-member set $\mathbb{D}_{\text{test}}^{\text{eval}}$, where$\mathbb{D}_{\text{test}}^{\text{atk}} \cap \mathbb{D}_{\text{test}}^{\text{eval}} = \varnothing$. Since $\mathbb{D}_{\text{r}}^{\text{atk}} \subset \mathbb{D}_{\text{r}}$, the forgetting samples are excluded from attack training by construction. We apply the same fixed attacker and candidate sets to θ_o, θ_u, and θ_{RE}.

Implementation on G-MIA attacker model: In all the experiments, G-MIA uses MLP with 128 hidden units and ReLU activation. We optimize it using Adam optimizer with learning rate 1e-3, batch size 64, and weight decay 1e-5 for at most 100 epochs. Model selection uses the best validation accuracy, and all configurations are fixed across datasets and target models.

Implementation on MIA verification: G-MIA vs. other MIAs. G-MIA provides an empirical audit rather than a certified deletion guarantee. We report the MIA scores on verifying VGG-13 / CIFAR-10 unlearning using different MIAs. As shown in Table 12, all MIAs show a consistent trend

towards different unlearning methods, but G-MIA is more sensitive and thereby being more suitable as an auditing tool.

Table 12: Independent membership audits on the original, retrained, and unlearned models. We report the AUC score and the Δ means (AUC(unl)-AUC(RE))/AUC(RE), where AUC(unl) is the AUC score of the unlearned corresponding method.

MIAs	Original	RE	FF-Erase	Δ	GA	Δ	(layer-wise) GA	Δ
G-MIA	$0.6315_{\pm 0.7\%}$	$0.5020_{\pm 0.9\%}$	$0.5135_{\pm 0.8\%}$	2.3%	$0.6287_{\pm 3.3\%}$	25.2%	$0.5027_{\pm 0.3\%}$	0.1%
FL	$0.5662_{\pm 0.6\%}$	$0.5245_{\pm 1.3\%}$	$0.5257_{\pm 0.6\%}$	0.2%	$0.5587_{\pm 3.1\%}$	6.5%	$0.5263_{\pm 0.3\%}$	0.3%
RMIA	$0.5633_{\pm 0.7\%}$	$0.5084_{\pm 0.9\%}$	$0.5083_{\pm 0.9\%}$	0.0%	$0.5601_{\pm 3.4\%}$	10.2%	$0.5096_{\pm 0.3\%}$	0.2%
LiRA	$0.6127_{\pm 0.9\%}$	$0.5017_{\pm 0.7\%}$	$0.5098_{\pm 0.7\%}$	1.6%	$0.6090_{\pm 2.9\%}$	21.4%	$0.5035_{\pm 0.2\%}$	0.4%
AMIA	$0.5882_{\pm 0.9\%}$	$0.5133_{\pm 1.0\%}$	$0.5340_{\pm 0.7\%}$	4.0%	$0.5713_{\pm 3.0\%}$	11.3%	$0.5207_{\pm 0.3\%}$	1.4%
CurvMIA	$0.6135_{\pm 0.6\%}$	$0.5128_{\pm 0.8\%}$	$0.5366_{\pm 0.7\%}$	4.6%	$0.5989_{\pm 3.0\%}$	16.8%	$0.5196_{\pm 0.3\%}$	1.3%
Grad-MIA	$0.6178_{\pm 0.8\%}$	$0.5044_{\pm 0.7\%}$	$0.5122_{\pm 0.9\%}$	1.5%	$0.6037_{\pm 3.1\%}$	19.7%	$0.5056_{\pm 0.2\%}$	0.2%
IHA	$0.6238_{\pm 0.7\%}$	$0.5042_{\pm 0.7\%}$	$0.5100_{\pm 0.6\%}$	1.2%	$0.6144_{\pm 3.2\%}$	21.9%	$0.5062_{\pm 0.3\%}$	0.4%
ImpMIA	$0.6269_{\pm 0.8\%}$	$0.5028_{\pm 1.1\%}$	$0.5193_{\pm 0.6\%}$	3.3%	$0.6148_{\pm 3.2\%}$	22.3%	$0.5035_{\pm 0.3\%}$	0.1%

Sensitivity Analysis on Temperature Scaling. To determine the implementation config of G-MIA, we explore the temperature hyperparameter T. As shown in Table 13, setting $T \geq 1.0$ degrades the attack effectiveness, as higher temperatures overly smooth the goodness distribution and dilute the overconfident goodness scores characteristic of training members. Conversely, applying aggressively small temperatures (*e.g.*, $T \leq 0.7$) also severely diminishes the auditing efficacy, with the AUC plummeting to 0.5516 at $T = 0.5$. In this highly sharpened regime, the scaled softmax function mathematically degenerates toward an argmax operation. This extreme compression forces the output toward a one-hot vector, thereby destroying the relative goodness distribution among non-target classes—a crucial source of structural membership signals. The optimal trade-off is located within $T \in [0.8, 0.9]$. Therefore, across all our experiments, $T = 0.85$ is fixed globally to sharpen the goodness distribution. This specific configuration yields the highest AUC (0.6315) and highly competitive ACC (0.6125) by amplifying the target-class confidence gap while successfully preserving the rich, continuous membership information embedded in the secondary classes. Therefore, we fix $T = 0.85$ for all G-MIA attacks reported in this paper.

Table 13: Ablation study on the temperature scaling factor T for G-MIA. We bold the best results and underline the second-best results.

Metric	0.5	0.6	0.7	0.8	0.85
ACC ↑	$0.5324_{\pm 0.8\%}$	$0.5819_{\pm 0.7\%}$	$0.6011_{\pm 0.6\%}$	$\mathbf{0.6127}_{\pm 0.7\%}$	$\underline{0.6125}_{\pm 0.7\%}$
AUC ↑	$0.5516_{\pm 0.7\%}$	$0.6006_{\pm 0.6\%}$	$0.6198_{\pm 0.5\%}$	$0.6298_{\pm 0.6\%}$	$\mathbf{0.6315}_{\pm 0.7\%}$
Metric	0.9	1.0	1.1	1.2	
ACC ↑	$0.6051_{\pm 0.7\%}$	$0.5847_{\pm 0.8\%}$	$0.5713_{\pm 0.7\%}$	$0.5508_{\pm 0.7\%}$	
AUC ↑	$\underline{0.6304}_{\pm 0.6\%}$	$0.6091_{\pm 0.7\%}$	$0.5890_{\pm 0.6\%}$	$0.5744_{\pm 0.7\%}$	

Implementation on G-MIA AUC score: We further examine whether the reduction in forget-set G-MIA AUC reflects selective unlearning or merely a global shift of the goodness features that invalidates the frozen attacker. We use the same G-MIA classifier trained on the original model and do not retrain or recalibrate it. Only the candidate sets used to evaluate its member scores are changed.

Let $\mathrm{AUC}_{f,t} = \mathrm{AUC}(\mathbb{D}_{\mathrm{f}}, \mathbb{D}_{\mathrm{test}})$, which is the primary unlearning-audit metric reported in the main text. We additionally report $\mathrm{AUC}_{r,t} = \mathrm{AUC}(\mathbb{D}_{\mathrm{r}}^{\mathrm{eval}}, \mathbb{D}_{\mathrm{test}}^{\mathrm{eval}})$, which tests whether the frozen attacker can still recognize retained training members after unlearning, and $\mathrm{AUC}_{r,f} = \mathrm{AUC}(\mathbb{D}_{\mathrm{r}}', \mathbb{D}_{\mathrm{f}})$, which measures the separability between retained and forgotten records. Here, $\mathbb{D}_{\mathrm{r}}' \subset \mathbb{D}_{\mathrm{r}}$ contains the same number of samples as $\mathbb{D}_{\mathrm{f}}$, and all evaluation subsets are disjoint from the data used to train G-MIA.

Throughout this paper, we report the orientation-normalized score AUC=$\max\{\ \mathrm{AUC}_{\mathrm{raw}}, 1-\mathrm{AUC}_{\mathrm{raw}}\}$, so that 0.5 denotes no distinguishability and larger values denote stronger separability. Selective unlearning should therefore produce the pattern $\mathrm{AUC}_{f,t} \approx 0.5, \quad \mathrm{AUC}_{r,t} > 0.5, \quad \mathrm{AUC}_{r,f} > 0.5$. In contrast, under-forgetting produces high $\mathrm{AUC}_{f,t}$ and low $\mathrm{AUC}_{r,f}$, whereas a global attacker failure or model collapse drives all three values toward 0.5.

Table 14: Calibration of the frozen G-MIA auditor on VGG-13/CIFAR-10. $\mathrm{AUC}_{f,t}$ evaluates forgotten versus unseen records, $\mathrm{AUC}_{r,t}$ evaluates retained versus unseen records, and $\mathrm{AUC}_{r,f}$ evaluates retained versus forgotten records. All AUCs are orientation-normalized; hence 0.5 denotes no distinguishability.

Method	$\mathrm{AUC}_{f,t} \to 0.5$	$\mathrm{AUC}_{r,t} \uparrow$	$\mathrm{AUC}_{r,f} \uparrow$
Original Model θ_o	$0.6317_{\pm 0.7\%}$	$0.6328_{\pm 0.6\%}$	$0.5013_{\pm 0.7\%}$
RE	$0.5020_{\pm 0.9\%}$	$0.6318_{\pm 0.6\%}$	$0.6320_{\pm 0.9\%}$
GA	$0.6287_{\pm 3.3\%}$	$0.6324_{\pm 0.7\%}$	$0.5026_{\pm 0.5\%}$
FYE	$0.6204_{\pm 2.9\%}$	$0.6305_{\pm 0.7\%}$	$0.5024_{\pm 0.8\%}$
PGU	$0.6277_{\pm 2.7\%}$	$0.6305_{\pm 0.6\%}$	$0.5035_{\pm 0.7\%}$
SCRUB	$0.6219_{\pm 2.1\%}$	$0.6308_{\pm 0.6\%}$	$0.5028_{\pm 0.7\%}$
BT	$0.6303_{\pm 3.3\%}$	$0.6318_{\pm 0.5\%}$	$0.5021_{\pm 0.9\%}$
ReGUn	$0.6351_{\pm 3.2\%}$	$0.6298_{\pm 0.8\%}$	$0.5023_{\pm 0.6\%}$
(layer-wise) GA	$0.5027_{\pm 0.3\%}$	$0.5033_{\pm 0.6\%}$	$0.5025_{\pm 0.4\%}$
(layer-wise) FYE	$0.5015_{\pm 0.8\%}$	$0.5021_{\pm 0.4\%}$	$0.5024_{\pm 0.6\%}$
(layer-wise) PGU	$0.5043_{\pm 0.7\%}$	$0.5028_{\pm 0.6\%}$	$0.5029_{\pm 0.6\%}$
(layer-wise) SCRUB	$0.5031_{\pm 0.7\%}$	$0.5029_{\pm 0.5\%}$	$0.5030_{\pm 0.4\%}$
(layer-wise) BT	$0.5210_{\pm 1.5\%}$	$0.5024_{\pm 0.5\%}$	$0.5021_{\pm 0.3\%}$
(layer-wise) ReGUn	$0.5312_{\pm 1.3\%}$	$0.5023_{\pm 0.6\%}$	$0.5023_{\pm 0.5\%}$
SISA	$0.5035_{\pm 0.9\%}$	$0.6345_{\pm 0.7\%}$	$0.6308_{\pm 0.7\%}$
FF-Erase	$0.5135_{\pm 0.8\%}$	$0.6290_{\pm 1.1\%}$	$0.6367_{\pm 0.9\%}$

As shown in Table 14, the original model exhibits high $\mathrm{AUC}_{f,t}$ and $\mathrm{AUC}_{r,t}$ but chance-level $\mathrm{AUC}_{r,f}$, as both forgetting and retained samples are training members. RE produces the desired selective-forgetting pattern: $\mathrm{AUC}_{f,t}$ decreases to 0.5020, while $\mathrm{AUC}_{r,t}$ remains 0.6318 and $\mathrm{AUC}_{r,f}$ increases to 0.6320. FF-Erase closely reproduces this pattern, obtaining $0.5135/0.6290/0.6367$ on the three metrics. In particular, the frozen attacker remains highly discriminative on retained members, ruling out the possibility that the decrease in forget-set AUC is merely caused by a global failure of the attacker after unlearning. Instead, the forgetting samples specifically become unseen-like while the retained samples preserve their membership behavior. The direct adaptations retain high $\mathrm{AUC}_{f,t}$ and $\mathrm{AUC}_{r,t}$ but chance-level $\mathrm{AUC}_{r,f}$, confirming under-forgetting. Conversely, the layer-wise adaptations drive all three metrics toward 0.5, showing that their low forget-set AUC results from global model collapse rather than selective removal. SISA exhibits the same selective pattern as retraining but at substantially higher computational cost. These results demonstrate that the G-MIA reduction of FF-Erase reflects forgetting-specific membership changes rather than a global goodness-distribution shift that invalidates the auditor.

C.4 Implementation on SOTA Unlearning Methods

Learning-Rate Sensitivity of Layer-wise Baselines. Table 15 examines whether the failure of layer-wise adaptations can be avoided by changing their learning rates over several orders of magnitude. Layer-wise GA exhibits a sharp transition: for $\eta \geq 10^{-5}$, all FF layers rapidly collapse to random-guess accuracy, whereas reducing η to 5×10^{-6} or below preserves utility but leaves both $\mathrm{Acc_f}$ and G-MIA close to the original model. Thus, no tested learning rate yields a retraining-like under/over-forgetting endpoint.

Layer-wise ReGUn exhibits the same trade-off more gradually. Large learning rates ($\eta \geq 5 \times 10^{-4}$) severely damage model utility, while intermediate rates improve removal only at the cost of substantial utility degradation. At $\eta \leq 5 \times 10^{-5}$, utility recovers, but $\mathrm{Acc_f}$ and G-MIA again approach the original-model behavior, indicating under-forgetting. The non-monotonic runtimes arise from the fixed early-stopping rule: configurations that either collapse rapidly or make negligible progress terminate earlier.

Table 15: Learning-rate sensitivity of representative layer-wise baselines on VGG-13/CIFAR-10 with 20% IID deletion. Red entries denote ineffective removal, impractical efficiency, or model collapse. Bold denotes the configuration reported in the main comparison.

Method	η	**Efficiency** t_{total} (s) ↓	**Effectiveness** Acc_f(%)→ RE	G-MIA → 0.5	**Utility** Acc_t(%)↑
Original Model θ_o	-	$1388.7_{\pm 2.0\%}$	$91.18_{\pm 0.5\%}$	$0.6317_{\pm 0.7\%}$	$82.17_{\pm 0.7\%}$
RE	-	$1107.0_{\pm 2.1\%}$	$79.26_{\pm 1.1\%}$	$0.5020_{\pm 0.9\%}$	$80.85_{\pm 0.8\%}$
FF-Erase	**1e-4**	$139.7_{\pm 3.6\%}$	$81.95_{\pm 1.0\%}$	$0.5135_{\pm 0.8\%}$	$79.48_{\pm 0.7\%}$
(layer-wise) GA	1e-3	$32.8_{\pm 3.3\%}$	$10.00_{\pm 0.0}$	$0.5035_{\pm 0.4\%}$	$10.00_{\pm 0.0}$
	1e-4	$35.8_{\pm 9.2\%}$	$10.00_{\pm 0.0}$	$0.5027_{\pm 0.3\%}$	$10.00_{\pm 0.0}$
	5e-5	$55.1_{\pm 8.6\%}$	$10.00_{\pm 0.0}$	$0.5022_{\pm 0.3\%}$	$10.00_{\pm 0.0}$
	1e-5	$284.0_{\pm 5.5\%}$	$10.00_{\pm 0.0}$	$0.5023_{\pm 0.2\%}$	$10.00_{\pm 0.0}$
	5e-6	$184.4_{\pm 4.2\%}$	$89.89_{\pm 1.2\%}$	$0.6289_{\pm 0.6\%}$	$81.95_{\pm 0.4\%}$
	1e-6	$76.8_{\pm 3.9\%}$	$89.92_{\pm 1.3\%}$	$0.6301_{\pm 0.6\%}$	$81.98_{\pm 0.4\%}$
	1e-7	$44.6_{\pm 2.1\%}$	$91.16_{\pm 0.3\%}$	$0.6311_{\pm 0.7\%}$	$82.19_{\pm 0.6\%}$
(layer-wise) ReGUn	5e-3	$37.5_{\pm 3.1\%}$	$10.00_{\pm 0.0}$	$0.5035_{\pm 0.4\%}$	$10.00_{\pm 0.0}$
	1e-3	$78.1_{\pm 8.3\%}$	$10.00_{\pm 0.0}$	$0.5033_{\pm 0.5\%}$	$10.00_{\pm 0.0}$
	5e-4	$94.2_{\pm 7.7\%}$	$27.55_{\pm 2.9\%}$	$0.5282_{\pm 0.9\%}$	$36.90_{\pm 1.1\%}$
	3e-4	$114.8_{\pm 6.5\%}$	$38.12_{\pm 2.7\%}$	$0.5312_{\pm 1.3\%}$	$41.08_{\pm 1.2\%}$
	1e-4	$208.3_{\pm 4.7\%}$	$77.35_{\pm 1.6\%}$	$0.5846_{\pm 0.9\%}$	$44.15_{\pm 1.1\%}$
	5e-5	$186.1_{\pm 3.3\%}$	$86.24_{\pm 2.3\%}$	$0.6245_{\pm 1.2\%}$	$69.11_{\pm 1.0\%}$
	1e-5	$89.4_{\pm 2.5\%}$	$91.09_{\pm 0.4\%}$	$0.6324_{\pm 0.4\%}$	$80.31_{\pm 0.3\%}$

These sweeps show that the reported failures are not attributable to one isolated learning-rate choice. Rather, the localized baseline objectives lack corresponding layer-specific goodness destinations and therefore do not exhibit a broad operating region that simultaneously approaches RE in removal effectiveness and retained utility. This result does not claim that every possible FF reformulation of GA or ReGUn must fail; it shows that learning-rate tuning alone does not resolve the mismatch.

D Additional G-MIA Experiment Results

Runtime protocol. All MIA implementations are evaluated under the same hardware and numerical-precision setting as the main experiments. We report $t_{\mathrm{sh.m}}$ as the total wall-clock cost of constructing the shadow/reference models required by each attack, and t_{mia} as the inference cost for verifying one target sample, excluding $t_{\mathrm{sh.m}}$. Query-based attacks use their prescribed numbers of target-model queries.

Table 16: MIA Comparison under VGG-13/CIFAR-100. We show the average results over 5 random seeds and report the mean value. The RSD for values in this table is below 1.4%.

Metric	G-MIA	FL	RMIA	LiRA	AMIA	CurvMIA	Grad-MIA	IHA	ImpMIA
ACC	**0.6184**	0.5496	0.5801	0.6015	0.5693	0.5875	0.6037	0.6125	0.6108
AUC	**0.6340**	0.5740	0.5709	0.6197	0.5920	0.6223	0.6268	0.6299	0.6312
d_{in}	5.2KB	**0.4KB**	**0.4KB**	**0.4KB**	**0.4KB**	**0.4KB**	471.2MB	421.4MB	421.4MB
$t_{\mathrm{sh.m}}$	**0s**	1316.3s	1303.5s	82664.2s	**0s**	**0s**	**0s**	**0s**	**0s**
t_{mia}	**6ms**	**6ms**	12 ms	0.66s	8.62s	4.17s	2.31s	8.92s	9.15s

Table 17: MIA Comparison under VGG-13/MNIST. We show the average results over 5 random seeds and report the mean value, with all RSDs below 0.5%.

Metric	G-MIA	FL	RMIA	LiRA	AMIA	CurvMIA	Grad-MIA	IHA	ImpMIA
ACC	**0.5960**	0.5393	0.5674	0.5668	0.5576	0.5641	0.5886	0.5864	0.5911
AUC	**0.6166**	0.5517	0.5889	0.5832	0.5780	0.5756	0.6016	0.6051	0.6080
d_{in}	0.52KB	**0.04KB**	**0.04KB**	**0.04KB**	**0.04KB**	**0.04KB**	458.8MB	408.8MB	408.1MB
$t_{\mathrm{sh.m}}$	**0s**	982.7s	981.3s	63104.3s	**0s**	**0s**	**0s**	**0s**	**0s**
t_{mia}	**6ms**	**6ms**	11 ms	0.63s	8.63s	4.14s	2.23s	8.81s	8.84s

We further evaluate G-MIA beyond the representative VGG-13/CIFAR-10 setting using CIFAR-100, MNIST, and Fashion-MNIST under the same attack protocol. As shown in Tables 16, 17,

Table 18: MIA Comparison under VGG-13/Fashion-MNIST. We show the average results over 5 random seeds and report the mean value. The RSD for values in this table is below 0.3%.

Metric	G-MIA	FL	RMIA	LiRA	AMIA	CurvMIA	Grad-MIA	IHA	ImpMIA
ACC	**0.7744**	0.6833	0.7117	0.7282	0.7382	0.7609	0.7507	0.7604	0.7596
AUC	**0.8217**	0.7138	0.7398	0.7408	0.7962	0.8150	0.7985	0.8107	0.8133
d_{in}	0.52KB	**0.04KB**	**0.04KB**	**0.04KB**	**0.04KB**	**0.04KB**	458.8MB	408.8MB	408.1MB
$t_{\text{sh.m}}$	**0s**	988.5s	980.6s	63093.8s	**0s**	**0s**	**0s**	**0s**	**0s**
t_{mia}	**6ms**	**6ms**	11 ms	0.61s	8.61s	4.01s	2.23s	8.77s	8.69s

and 18, G-MIA achieves the highest mean ACC and AUC across all three benchmarks while retaining substantially lower verification overhead. In particular, G-MIA requires no shadow-model training, uses only 0.52–5.2 KB of layer-wise goodness inputs, and completes a single inference in approximately 6 ms. In contrast, shadow-model-based attacks incur substantial additional training cost, while perturbation- and white-box attacks require either repeated queries or MB-scale internal information. These results show that G-MIA consistently provides a favorable accuracy-efficiency trade-off across datasets, rather than relying on a particular benchmark.

E FF Unlearning under Practical Deletion Requests

Beyond IID sample deletion, we evaluate FF-Erase under five practical deletion requests on VGG-13/CIFAR-10: class-wise unlearning, backdoor removal, mislabeled-sample forgetting, non-IID deletion, and high-influence data erasure. For compactness, we report GA Sekhari et al. (2021) and SCRUB Kurmanji et al. (2023) as representative gradient-based and distillation-based baselines, together with their FF-adapted layer-wise variants. For all experiments, G-MIA is trained once on the original model θ_o and remains fixed when evaluating the unlearned models.

Class-wise Unlearning. A class-level deletion request removes an entire class of the $\mathbb{D}_{\text{train}}$ rather than an IID subset, leaving no retained examples from which the model can preserve that class. In this experiment, we remove all airplane training samples (the first class in CIFAR-10 dataset), retain the original ten-class output space, and exclude the target class from the guidance set. Utility is evaluated on test samples from the remaining nine classes. As shown in Table 19, FF-Erase reduces target-class accuracy to 3.46%, approaching RE's 0.00%, while retaining 78.7% accuracy on the remaining classes and achieving a G-MIA AUC of 0.5092. In contrast, direct GA and SCRUB retain over 73% target-class accuracy and substantial membership signals, whereas their layer-wise extensions collapse. FF-Erase completes the request in 142.6 s, yielding an 8.7× speedup over RE.

Table 19: Class-wise unlearning. For efficiency, we show the total time for unlearning t_{total}; for removal effectiveness, we present the model accuracy on target class Acc_{cls} and G-MIA AUC; for utility, we present the model accuracy on test set excluded unlearned class Acc'_{t}. We mark the ineffective removal and model collapse in red.

Method	t_{total} (s) ↓	Acc_{cls} (%) → RE	Acc'_{t} (%) ↑	G-MIA AUC↓
RE	$1238.3_{\pm 2.4\%}$	$0.00_{\pm 0.0}$	$81.02_{\pm 1.3\%}$	$0.5061_{\pm 1.1\%}$
GA	$289.4_{\pm 3.7\%}$	$73.64_{\pm 3.1\%}$	$76.15_{\pm 2.3\%}$	$0.5978_{\pm 1.4\%}$
SCRUB	$415.3_{\pm 2.5\%}$	$74.77_{\pm 3.3\%}$	$76.63_{\pm 2.9\%}$	$0.6013_{\pm 1.7\%}$
(layer-wise) GA	$38.1_{\pm 5.3\%}$	$10.00_{\pm 0.0\%}$	$11.11_{\pm 0.0\%}$	$0.5106_{\pm 1.4\%}$
(layer-wise) SCRUB	$96.6_{\pm 4.7\%}$	$10.00_{\pm 0.0\%}$	$11.11_{\pm 0.1\%}$	$0.5133_{\pm 1.1\%}$
FF-Erase	$142.6_{\pm 2.7\%}$	$3.46_{\pm 1.6\%}$	$78.75_{\pm 2.4\%}$	$0.5092_{\pm 1.3\%}$

Backdoor removal. Backdoor unlearning aims to erase a malicious trigger–target association while preserving clean predictions. We poison 5% of the training samples by inserting a 4×4 white-square trigger at the bottom-right corner and relabeling them to target class 0. The poisoned-sample identities are provided by the deletion request. Removal effectiveness is evaluated using the attack success rate (ASR) on triggered non-target test samples, while utility is measured on the clean test set. As shown in Table 20, FF-Erase reduces ASR from 96.36% to 5.35%, while retaining 79.45% clean accuracy and reducing G-MIA AUC to 0.5093. Direct GA and SCRUB leave ASR above 91%, whereas their layer-wise variants reduce ASR only by collapsing clean accuracy to the random-guess level. FF-Erase requires 136.7 s, achieving a 9.6× speedup over RE.

Table 20: Backdoor removal. For efficiency, we show the total time for unlearning t_{total} (for θ_o, it refers to training time); for removal effectiveness, we present the ASR of backdoor attack and G-MIA AUC; for utility, we present the model accuracy on test set Acc_t which is clean. We mark the ineffective removal and model collapse in red.

Method	t_{total} (s) ↓	ASR (%) ↓	Acc_t (%) ↑	G-MIA AUC↓
Original model θ_o	$1382.5_{\pm 2.0\%}$	$96.36_{\pm 1.5\%}$	$82.09_{\pm 1.1\%}$	$0.6301_{\pm 1.1\%}$
RE	$1315.2_{\pm 2.1\%}$	$0.00_{\pm 0.0}$	$80.72_{\pm 1.3\%}$	$0.5061_{\pm 1.1\%}$
GA	$287.7_{\pm 3.3\%}$	$91.07_{\pm 3.0\%}$	$76.11_{\pm 3.3\%}$	$0.6279_{\pm 1.3\%}$
SCRUB	$418.1_{\pm 2.5\%}$	$92.71_{\pm 3.4\%}$	$76.49_{\pm 3.4\%}$	$0.6233_{\pm 1.6\%}$
(layer-wise) GA	$42.1_{\pm 3.7\%}$	$10.11_{\pm 1.2\%}$	$10.00_{\pm 0.0}$	$0.5121_{\pm 1.4\%}$
(layer-wise) SCRUB	$76.3_{\pm 3.5\%}$	$10.11_{\pm 1.1\%}$	$10.00_{\pm 0.0}$	$0.5083_{\pm 1.2\%}$
FF-Erase	$136.7_{\pm 2.3\%}$	$5.35_{\pm 1.3\%}$	$79.45_{\pm 0.4\%}$	$0.5093_{\pm 1.3\%}$

Mislabeled-sample Forgetting. We randomly assign incorrect labels to 10% of the training samples to construct the mislabeled subset $\mathbb{D}_{mis} \subset \mathbb{D}_{train}$. The deletion request identifies the affected samples; FF-Erase requires only their identities and does not use either the corrupted or corrected labels. For baselines whose objectives require labels, the corresponding corrupted labels are additionally provided. We construct the RE reference by retraining from scratch after restoring the affected samples to their clean labels, which are also used to evaluate Acc_{mis}. As shown in Table 21, FF-Erase recovers clean-label accuracy from 63.22% in the original model to 77.89%, approaching RE's 81.06%, while preserving 80.68% test accuracy and achieving a G-MIA AUC of 0.5077. Direct GA and SCRUB fail to recover the corrected behavior and retain substantial membership information, while their layer-wise variants collapse. FF-Erase completes unlearning in 143.6 s, an 8.6× speedup over RE.

Table 21: Mislabeled-sample forgetting. For efficiency, we show the total time for unlearning t_{total} (for θ_o, it refers to training time); for removal effectiveness, we present the model accuracy on mislabeled samples Acc_{mis} and G-MIA AUC; for utility, we present the model accuracy on test set Acc_t. We mark the ineffective removal and model collapse in red.

Method	t_{total} (s) ↓	Acc_{mis} (%) ↑	Acc_t (%) ↑	G-MIA AUC↓
Original model θ_o	$1382.5_{\pm 2.2\%}$	$63.22_{\pm 1.7\%}$	$77.99_{\pm 0.7\%}$	$0.6351_{\pm 1.0\%}$
RE	$1241.7_{\pm 2.9\%}$	$81.06_{\pm 0.5\%}$	$81.57_{\pm 1.0\%}$	$0.5053_{\pm 1.1\%}$
GA	$287.7_{\pm 3.3\%}$	$58.32_{\pm 3.3\%}$	$71.78_{\pm 2.5\%}$	$0.6181_{\pm 1.4\%}$
SCRUB	$411.9_{\pm 2.0\%}$	$59.71_{\pm 3.4\%}$	$70.49_{\pm 3.4\%}$	$0.6156_{\pm 1.9\%}$
(layer-wise) GA	$44.1_{\pm 3.2\%}$	$10.00_{\pm 0.0}$	$10.00_{\pm 0.0}$	$0.5133_{\pm 1.1\%}$
(layer-wise) SCRUB	$92.3_{\pm 3.3\%}$	$10.00_{\pm 0.0}$	$10.00_{\pm 0.0}$	$0.5085_{\pm 0.9\%}$
FF-Erase	$143.6_{\pm 2.0\%}$	$77.89_{\pm 3.4\%}$	$80.68_{\pm 1.4\%}$	$0.5077_{\pm 1.1\%}$

Distribution-shifted (non-IID) deletion. The forgetting data $\mathbb{D}_f$ can be a non-IID distribution from the overall training set, *e.g.*, federated unlearning. To this end, we construct a strongly label-skewed forget set containing 20% of the training samples using a Dirichlet partition with concentration parameter $\alpha = 0.1$. As shown in Table 22, FF-Erase closely reproduces the RE behavior despite the distribution shift: Acc_f differs by only 0.20 percentage points (77.36% vs. 77.16%), test accuracy by 0.30 points (80.03% vs. 80.33%), and G-MIA AUC by 0.0027 (0.5063 vs. 0.5036). Direct GA and SCRUB retain substantial residual membership information, whereas their layer-wise extensions collapse. FF-Erase requires 139.1 s compared with 1113.5 s for RE, yielding an 8.0× speedup.

High-influence data erasure. High-influence samples can disproportionately affect the learned decision boundary and therefore constitute a challenging deletion request. Erasing those data points are heavily affecting the model performance. We adapt TracIn Pruthi et al. (2020) to FF by computing its influence separately under each layer-wise local objective and averaging across the L layers, $I(\boldsymbol{x}) = 1/L \sum_{l=1}^{L} I^l(\boldsymbol{x})$, and select the top 1% of training samples according to this score as $\mathbb{D}_{high-inf}$. As shown in Table 23, FF-Erase reduces accuracy on the selected high-influence samples from 89.75% to 59.32%, approaching RE's 51.44%, while retaining 78.55% test accuracy and reducing G-MIA AUC from 0.6377 to 0.5289. Direct GA and SCRUB barely remove the influence of these samples, whereas their layer-wise variants again collapse. Although this challenging request

Table 22: Distribution-shifted (non-IID) deletion. For efficiency, we show the total time for unlearning t_{total} (for θ_o, it refers to training time); for removal effectiveness, we present the model accuracy on forgetting data Acc_f and G-MIA AUC; for utility, we present the model accuracy on test set Acc_t. We mark the ineffective removal and model collapse in red.

Method	t_{total} (s) ↓	Acc_f (%) ↑	Acc_t (%) ↑	G-MIA AUC↓
Original model θ_o	$1386.0_{\pm 2.2\%}$	$91.19_{\pm 0.7\%}$	$82.20_{\pm 0.9\%}$	$0.6343_{\pm 0.9\%}$
RE	$1113.5_{\pm 2.1\%}$	$77.16_{\pm 2.3\%}$	$80.33_{\pm 1.1\%}$	$0.5036_{\pm 0.9\%}$
GA	$281.8_{\pm 3.3\%}$	$87.23_{\pm 3.2\%}$	$78.98_{\pm 2.2\%}$	$0.6307_{\pm 1.3\%}$
SCRUB	$413.3_{\pm 2.0\%}$	$86.90_{\pm 3.6\%}$	$78.79_{\pm 3.6\%}$	$0.6269_{\pm 2.2\%}$
(layer-wise) GA	$45.2_{\pm 3.1\%}$	$10.00_{\pm 0.0}$	$10.00_{\pm 0.0}$	$0.5098_{\pm 1.0\%}$
(layer-wise) SCRUB	$96.1_{\pm 3.0\%}$	$10.00_{\pm 0.0}$	$10.00_{\pm 0.0}$	$0.5078_{\pm 0.9\%}$
FF-Erase	$139.1_{\pm 2.3\%}$	$77.36_{\pm 3.0\%}$	$80.03_{\pm 1.3\%}$	$0.5063_{\pm 1.5\%}$

leaves a larger gap to RE than the preceding settings, FF-Erase achieves substantially stronger removal without sacrificing model utility and requires only 138.4 s, yielding a 9.6× speedup over RE.

Table 23: High-influence data erasure. For efficiency, we show the total time for unlearning t_{total} (for θ_o, it refers to training time); for removal effectiveness, we present the model accuracy on selected high-influence data subset Acc_h and G-MIA AUC; for utility, we present the model accuracy on test set Acc_t which is clean. We mark the ineffective removal and model collapse in red.

Method	t_{total} (s) ↓	Acc_h (%) ↓	Acc_t (%) ↑	**G-MIA AUC↓**
Original model θ_o	$1388.7_{\pm 2.0\%}$	$89.75_{\pm 0.8\%}$	$82.17_{\pm 0.7\%}$	$0.6377_{\pm 0.8\%}$
RE	$1330.5_{\pm 2.7\%}$	$51.44_{\pm 4.6\%}$	$77.72_{\pm 1.2\%}$	$0.5037_{\pm 1.0\%}$
GA	$221.0_{\pm 3.2\%}$	$88.67_{\pm 3.9\%}$	$79.12_{\pm 3.8\%}$	$0.6303_{\pm 1.1\%}$
SCRUB	$249.8_{\pm 2.1\%}$	$88.74_{\pm 3.2\%}$	$79.33_{\pm 3.9\%}$	$0.6319_{\pm 1.2\%}$
(layer-wise) GA	$37.7_{\pm 3.1\%}$	$10.00_{\pm 0.0\%}$	$10.00_{\pm 0.0\%}$	$0.5109_{\pm 1.1\%}$
(layer-wise) SCRUB	$98.5_{\pm 3.1\%}$	$10.00_{\pm 0.0\%}$	$10.00_{\pm 0.0\%}$	$0.5121_{\pm 1.2\%}$
FF-Erase	$138.4_{\pm 2.1\%}$	$59.32_{\pm 3.8\%}$	$78.55_{\pm 1.4\%}$	$0.5289_{\pm 2.2\%}$

Overall. Across all five practical deletion requests, FF-Erase remains stable and achieves an 8.0×–9.6× speedup over retraining while consistently moving the model toward the corresponding RE behavior. Direct BP-oriented methods largely preserve utility but leave substantial residual information, whereas naïvely extending them to every FF layer frequently produces apparently low G-MIA scores only through model collapse. These results demonstrate that the guidance-driven, layer-specific goodness realignment of FF-Erase generalizes beyond IID sample deletion to heterogeneous and practically relevant forgetting requests. It is also noted that the largest residual gap occurs for high-influence deletion, where the forget set is explicitly selected to maximize estimated training influence. This setting requires larger parameter changes than random deletion and creates a more difficult under/over-forgetting trade-off. Similarly, utility degradation becomes more pronounced at a 50% deletion ratio. These results delimit the current operating regime of FF-Erase rather than indicating uniform equivalence to retraining.

F Scalability across Unlearning Ratio

As shown in Table 24, we further evaluate the scalability of FF-Erase under different unlearning ratios $\delta \in \{5\%, 10\%, 20\%, 30\%, 50\%\}$ on VGG-13/CIFAR-10 with IID deletion. Across all ratios, FF-Erase maintains a nearly stable end-to-end cost of 123.2–143.0s, while retraining requires 691.4–1311.2s, yielding a 5.6×–9.2× speedup. More importantly, its removal behavior consistently tracks the corresponding retraining reference: the gap in ACC_f remains within 2.69 percentage points, while G-MIA stays close to the retrained model with an absolute gap of at most 0.0169. The retained utility also remains close to RE for deletion ratios up to 30%, with only 0.94–1.60 percentage-point degradation. Even under the challenging 50% deletion setting, FF-Erase preserves effective removal while remaining (5.6×) faster than retraining, although the utility gap increases to 3.99 points. Overall, these results show that FF-Erase scales robustly across substantially different deletion ratios without requiring ratio-specific algorithmic changes.

Table 24: FF-Erase performance under different unlearning ratio δ under VGG-13 model, CIFAR-10 dataset, and IID unlearning task. For assessing the metrics, we also provide the time of retraining as t_{RE}, Acc_f/Acc_t/G-MIA of RE baseline as Acc_f^*/Acc_t^*/G-MIA*. For all cases, t_0=57.1$_{\pm 2.3\%}$ s.

δ	Efficiency			Effectiveness				Utility	
	t_1 (s) ↓	t_{total} (s) ↓	t_{RE} (s)	Acc_f(%)	Acc_f^*(%)	G-MIA ↓	G-MIA*	Acc_t(%)	Acc_t^*(%)
5%	84.9$_{\pm 2.9\%}$	142.0$_{\pm 3.5\%}$	1311.2$_{\pm 1.9\%}$	82.97$_{\pm 1.4\%}$	81.03$_{\pm 1.3\%}$	0.5125$_{\pm 0.7\%}$	0.5034$_{\pm 1.0\%}$	80.72$_{\pm 0.8\%}$	81.66$_{\pm 0.9\%}$
10%	85.9$_{\pm 2.7\%}$	143.0$_{\pm 3.3\%}$	1257.8$_{\pm 1.9\%}$	82.51$_{\pm 1.1\%}$	80.75$_{\pm 0.9\%}$	0.5187$_{\pm 0.8\%}$	0.5018$_{\pm 1.1\%}$	79.91$_{\pm 0.7\%}$	81.19$_{\pm 0.8\%}$
20%	82.6$_{\pm 2.9\%}$	139.7$_{\pm 3.6\%}$	1107.0$_{\pm 2.1\%}$	81.95$_{\pm 1.0\%}$	79.26$_{\pm 1.1\%}$	0.5135$_{\pm 0.8\%}$	0.5020$_{\pm 0.9\%}$	79.48$_{\pm 0.7\%}$	80.85$_{\pm 0.8\%}$
30%	80.3$_{\pm 3.1\%}$	137.4$_{\pm 3.8\%}$	972.7$_{\pm 2.2\%}$	78.78$_{\pm 1.3\%}$	77.37$_{\pm 1.2\%}$	0.5189$_{\pm 0.7\%}$	0.5085$_{\pm 0.9\%}$	76.27$_{\pm 0.8\%}$	77.87$_{\pm 0.9\%}$
50%	66.1$_{\pm 3.3\%}$	123.2$_{\pm 3.9\%}$	691.4$_{\pm 1.9\%}$	71.08$_{\pm 1.2\%}$	70.14$_{\pm 0.9\%}$	0.5113$_{\pm 0.9\%}$	0.5044$_{\pm 1.1\%}$	65.13$_{\pm 1.2\%}$	69.12$_{\pm 0.7\%}$

G Sequential Unlearning Requests

Table 25: Sequential unlearning on VGG-13/CIFAR-10. Each stage removes a new disjoint 5% subset of the training data. t_{total} denotes the cumulative sequential cost with the Val-Train guidance cost counted only once; "one" denotes the corresponding one-shot FF-Erase result at the same cumulative deletion ratio.

Stage	Cumulative δ	Efficiency		Removal Accuracy		G-MIA		Utility	
		t_1 (s)	t_{total} (s)	Acc_f (%)	Acc_f^{one} (%)	G-MIA	G-MIA^{one}	Acc_t (%)	Acc_t^{one} (%)
1	5%	84.9$_{\pm 2.7\%}$	142.0$_{\pm 3.1\%}$	82.97$_{\pm 1.9\%}$	82.97$_{\pm 1.4\%}$	0.5123$_{\pm 0.7\%}$	0.5125$_{\pm 0.7\%}$	80.71$_{\pm 0.7\%}$	80.72$_{\pm 0.9\%}$
2	10%	85.9$_{\pm 1.7\%}$	227.9$_{\pm 3.7\%}$	81.31$_{\pm 1.3\%}$	82.51$_{\pm 1.1\%}$	0.5179$_{\pm 0.9\%}$	0.5187$_{\pm 0.8\%}$	80.12$_{\pm 0.5\%}$	79.91$_{\pm 0.8\%}$
3	15%	81.7$_{\pm 1.5\%}$	309.6$_{\pm 4.0\%}$	81.77$_{\pm 1.6\%}$	–	0.5101$_{\pm 0.8\%}$	–	79.91$_{\pm 0.8\%}$	–
4	20%	74.6$_{\pm 1.9\%}$	384.2$_{\pm 4.2\%}$	80.41$_{\pm 1.4\%}$	81.95$_{\pm 1.0\%}$	0.5089$_{\pm 0.7\%}$	0.5135$_{\pm 0.8\%}$	79.85$_{\pm 0.4\%}$	79.48$_{\pm 0.8\%}$
5	25%	67.2$_{\pm 1.8\%}$	451.4$_{\pm 4.6\%}$	79.03$_{\pm 1.5\%}$	–	0.5074$_{\pm 0.8\%}$	–	78.16$_{\pm 0.7\%}$	–
6	30%	63.1$_{\pm 1.2\%}$	514.5$_{\pm 4.8\%}$	77.76$_{\pm 1.7\%}$	78.78$_{\pm 1.3\%}$	0.5066$_{\pm 0.6\%}$	0.5189$_{\pm 0.7\%}$	77.40$_{\pm 0.7\%}$	76.27$_{\pm 0.9\%}$

Real-world deletion requests may arrive sequentially rather than being known in advance. To evaluate this setting, we partition 30% of the training data into six disjoint deletion subsets $\{\mathbb{D}_{\mathrm{f}}^{(k)}\}_{k=1}^{6}$, each containing 5% of the training samples. Starting from the original model, Stage k applies FF-Erase to the new request $\mathbb{D}_{\mathrm{f}}^{(k)}$ using the model returned by Stage $k-1$, such that the cumulative forget set after Stage k is $\mathbb{D}_{\mathrm{f}}^{1:k} = \bigcup_{i=1}^{k} \mathbb{D}_{\mathrm{f}}^{(i)}$. The Val-Train guidance model is trained only once and reused across all requests. Consequently, the cumulative cost after k requests is $t_{\mathrm{total}}^{(k)} = t_0 + \sum_{i=1}^{k} t_1^{(i)}$, where $t_0 = 57.1$ s is counted only once. For cumulative deletion ratios available in §F, we additionally report the corresponding one-shot FF-Erase results for comparison.

As shown in Table 25, FF-Erase remains stable throughout all six sequential deletion requests. At the matched cumulative ratios of 5%, 10%, 20%, and 30%, sequential unlearning achieves removal and utility comparable to the corresponding one-shot runs. In particular, after six requests (30% cumulative deletion), sequential FF-Erase obtains 77.76% Acc_f, 0.5066 G-MIA, and 77.40% Acc_t, compared with 78.78%, 0.5189, and 76.27% for one-shot deletion. Sequential updates do not introduce additional degradation relative to the matched one-shot runs over the evaluated six-request sequence.

The cumulative runtime naturally increases with the number of requests because each newly arriving request requires an additional unlearning update. Nevertheless, the Val-Train guidance model is constructed only once and reused throughout the sequence. After six requests, the complete sequential procedure requires 514.5 s, which remains 1.9× faster than retraining once on the final 70% retained dataset (972.7 s). Moreover, the per-request update cost does not increase over time, decreasing from approximately 84.9 s at Stage 1 to 63.1 s at Stage 6. These results demonstrate that FF-Erase supports repeated deletion requests without retraining the guidance model or suffering progressive model collapse.

H Layer-wise Unlearning Verification using CKA Scores

As FF models are trained through layer-wise greedy optimization, different layers may retain different aspects of knowledge from the training data. To examine whether FF-Erase induces layer-wise representation changes comparable to retraining, we further evaluate its layer-wise unlearning behavior using Centered Kernel Alignment (CKA) similarity Wang et al. (2015); Kornblith et al. (2019).

CKA is a similarity measure between two sets of representations and has been widely used to analyze layer-wise representation changes in machine unlearning Xu et al. (2025); Kim et al. (2025). Given two representation matrices $X_i^o \in \mathbb{R}^{n\times p}$ from the original model and $X_i^u \in \mathbb{R}^{n\times p}$ from the unlearned model, where n is the number of samples and p is the dimensionality of the representation at layer i, their linear CKA similarity is calculated as:

$$\mathrm{CKA}(X_i^o, X_i^u) = \frac{\mathrm{HSIC}(X_i^o, X_i^u)}{\sqrt{\mathrm{HSIC}(X_i^o, X_i^o)\mathrm{HSIC}(X_i^u, X_i^u)}}, \tag{6}$$

where $\mathrm{HSIC}(\cdot,\cdot)$ is the Hilbert-Schmidt Independence Criterion. Detailed calculations of HSIC can be found in Wang et al. (2015). CKA ranges from 0 to 1, where a lower value indicates a larger difference between two representations.

Following the setting of Table 3, we compute CKA on the forgetting data $\mathbb{D}_\mathrm{f}$ between the original model and models produced by FF-Erase, retraining (RE), direct BP-oriented baselines, and their layer-wise adaptations. We use RE as the reference for the layer-wise representation changes expected after unlearning. The results are reported in Table 26.

Table 26: Layer-wise CKA similarity between the original and unlearned models on $\mathbb{D}_\mathrm{f}$ under VGG-13/CIFAR-10. We use ↑ (↓) to denote CKA values more than 20% above (below) the corresponding RE reference, indicating insufficient (excessive) representation changes, respectively.

Layer	1	2	3	4	5	6	7
RE	$0.9918_{\pm 0.5\%}$	$0.9249_{\pm 0.9\%}$	$0.8004_{\pm 1.1\%}$	$0.6319_{\pm 1.0\%}$	$0.6286_{\pm 1.3\%}$	$0.5700_{\pm 1.2\%}$	$0.3619_{\pm 1.3\%}$
FF-Erase	$0.9961_{\pm 0.0\%}$	$0.9421_{\pm 1.9\%}$	$0.8099_{\pm 3.3\%}$	$0.8279_{\pm 2.7\%}$ ↑	$0.6643_{\pm 1.9\%}$	$0.7454_{\pm 2.3\%}$ ↑	$0.4192_{\pm 2.1\%}$
GA	$1.0000_{\pm 0.0\%}$	$1.0000_{\pm 0.0\%}$	$1.0000_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑
FYE	$1.0000_{\pm 0.0\%}$	$1.0000_{\pm 0.0\%}$	$1.0000_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑
PGU	$1.0000_{\pm 0.0\%}$	$1.0000_{\pm 0.0\%}$	$1.0000_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑
SCRUB	$1.0000_{\pm 0.0\%}$	$1.0000_{\pm 0.0\%}$	$1.0000_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑
BT	$1.0000_{\pm 0.0\%}$	$1.0000_{\pm 0.0\%}$	$1.0000_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑
ReGUn	$1.0000_{\pm 0.0\%}$	$1.0000_{\pm 0.0\%}$	$1.0000_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑
(layer-wise) GA	$0.5811_{\pm 3.5\%}$ ↓	$0.3812_{\pm 3.1\%}$ ↓	$0.2840_{\pm 3.7\%}$ ↓	$0.2545_{\pm 2.9\%}$ ↓	$0.2810_{\pm 3.2\%}$ ↓	$0.1877_{\pm 3.4\%}$ ↓	$0.1527_{\pm 3.6\%}$ ↓
(layer-wise) FYE	$0.6315_{\pm 3.2\%}$ ↓	$0.4267_{\pm 3.4\%}$ ↓	$0.3531_{\pm 3.1\%}$ ↓	$0.3907_{\pm 3.3\%}$ ↓	$0.3236_{\pm 3.7\%}$ ↓	$0.2826_{\pm 2.9\%}$ ↓	$0.2133_{\pm 3.3\%}$ ↓
(layer-wise) PGU	$0.6335_{\pm 3.1\%}$ ↓	$0.4209_{\pm 3.2\%}$ ↓	$0.3266_{\pm 3.3\%}$ ↓	$0.3799_{\pm 3.1\%}$ ↓	$0.3810_{\pm 2.6\%}$ ↓	$0.3278_{\pm 2.5\%}$ ↓	$0.2553_{\pm 3.7\%}$ ↓
(layer-wise) SCRUB	$0.8813_{\pm 3.2\%}$	$0.9108_{\pm 3.3\%}$	$0.4512_{\pm 3.3\%}$ ↓	$0.3861_{\pm 3.3\%}$ ↓	$0.3352_{\pm 3.1\%}$ ↓	$0.3525_{\pm 3.0\%}$ ↓	$0.2521_{\pm 2.0\%}$ ↓
(layer-wise) BT	$0.8965_{\pm 3.1\%}$	$0.8560_{\pm 3.0\%}$	$0.5650_{\pm 3.3\%}$ ↓	$0.3996_{\pm 2.7\%}$ ↓	$0.3623_{\pm 2.8\%}$ ↓	$0.3898_{\pm 2.8\%}$ ↓	$0.1594_{\pm 2.7\%}$ ↓
(layer-wise) ReGUn	$0.9212_{\pm 3.2\%}$	$0.8973_{\pm 3.6\%}$	$0.6819_{\pm 3.5\%}$	$0.3453_{\pm 3.1\%}$ ↓	$0.4826_{\pm 3.4\%}$ ↓	$0.3582_{\pm 3.0\%}$ ↓	$0.3215_{\pm 2.9\%}$
Layer	8	9	10	11	12	13	-
RE	$0.3472_{\pm 1.3\%}$	$0.4981_{\pm 1.3\%}$	$0.4200_{\pm 1.1\%}$	$0.4923_{\pm 1.4\%}$	$0.6223_{\pm 1.2\%}$	$0.6144_{\pm 1.3\%}$	-
FF-Erase	$0.3919_{\pm 2.3\%}$	$0.5037_{\pm 1.9\%}$	$0.4128_{\pm 1.3\%}$	$0.4789_{\pm 2.2\%}$	$0.6308_{\pm 2.5\%}$	$0.6158_{\pm 2.7\%}$	-
GA	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.1642_{\pm 4.7\%}$ ↓	-
FYE	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.1854_{\pm 3.3\%}$ ↓	-
PGU	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.5854_{\pm 2.3\%}$	-
SCRUB	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.2333_{\pm 4.2\%}$ ↓	-
BT	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.2988_{\pm 3.1\%}$ ↓	-
ReGUn	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.9999_{\pm 0.0\%}$ ↑	$0.6356_{\pm 1.7\%}$	-
(layer-wise) GA	$0.1523_{\pm 3.3\%}$ ↓	$0.1623_{\pm 3.1\%}$ ↓	$0.1122_{\pm 2.0\%}$ ↓	$0.1115_{\pm 2.9\%}$ ↓	$0.0892_{\pm 3.3\%}$ ↓	$0.0656_{\pm 3.0\%}$ ↓	-
(layer-wise) FYE	$0.2352_{\pm 3.1\%}$ ↓	$0.1898_{\pm 3.2\%}$ ↓	$0.1510_{\pm 3.0\%}$ ↓	$0.0989_{\pm 2.7\%}$ ↓	$0.0942_{\pm 3.0\%}$ ↓	$0.0516_{\pm 2.7\%}$ ↓	-
(layer-wise) PGU	$0.3837_{\pm 3.0\%}$	$0.3089_{\pm 3.4\%}$ ↓	$0.2894_{\pm 2.9\%}$ ↓	$0.2288_{\pm 2.9\%}$ ↓	$0.1827_{\pm 2.6\%}$ ↓	$0.1834_{\pm 2.9\%}$ ↓	-
(layer-wise) SCRUB	$0.4521_{\pm 3.1\%}$ ↑	$0.4630_{\pm 3.3\%}$	$0.5281_{\pm 2.8\%}$ ↑	$0.3819_{\pm 3.5\%}$ ↓	$0.8916_{\pm 3.0\%}$ ↑	$0.7442_{\pm 3.3\%}$ ↑	-
(layer-wise) BT	$0.5818_{\pm 3.1\%}$ ↑	$0.5662_{\pm 3.5\%}$	$0.6049_{\pm 2.8\%}$ ↑	$0.4008_{\pm 2.7\%}$	$0.8727_{\pm 2.6\%}$ ↑	$0.6177_{\pm 2.3\%}$	-
(layer-wise) ReGUn	$0.4036_{\pm 3.3\%}$	$0.5898_{\pm 2.1\%}$	$0.5864_{\pm 2.0\%}$ ↑	$0.4315_{\pm 2.9\%}$	$0.7896_{\pm 3.3\%}$ ↑	$0.6236_{\pm 2.9\%}$	-

Retraining reveals strongly layer-dependent representation changes. As shown in Table 26, RE exhibits high CKA in the shallow layers, substantially lower similarity in the middle layers, and moderately increased similarity toward the deepest layers. This non-monotonic profile indicates that removing the same training samples affects different FF layers to substantially different degrees. In particular, the high similarity of the first layers is consistent with their extraction of generic features shared across training samples, whereas the larger changes in intermediate layers suggest greater sensitivity to the removed samples. We therefore use the complete layer-wise CKA profile of RE, rather than uniformly lower CKA, as the unlearning reference.

FF-Erase most closely reproduces the retraining-induced representation changes. Across the 13 layers, the CKA profile of FF-Erase closely follows RE, with 11 layers remaining within 20% of the corresponding retraining reference. The largest residual differences occur at Layers 4 and 6, where FF-Erase retains moderately higher similarity to the original model. Nevertheless, its remaining layers closely match RE, including the deep representations in Layers 9–13. This result complements the prediction- and membership-level evaluation by showing that FF-Erase induces retraining-like representation changes throughout the FF hierarchy rather than only modifying the final prediction behavior.

BP-oriented baselines expose two distinct failure modes on FF models. Under the direct adaptation, GA, FYE, PGU, SCRUB, BT, and ReGUn update only the FF prediction endpoint.Consequently,

their CKA values remain essentially 1.0 throughout Layers 1–12, showing that their intermediate representations remain unchanged. Although the final-layer representation can change substantially, modifying only this endpoint cannot reproduce the layer-wise effect of retraining. Conversely, independently extending these objectives to every FF layer frequently over-perturbs the representations: layer-wise GA and FYE fall far below the RE profile at all 13 layers, while layer-wise PGU exhibits the same behavior in 12 of 13 layers, consistent with their severe utility degradation and model collapse.

More structured layer-wise objectives mitigate, but do not eliminate, cross-layer mismatch. Layer-wise SCRUB, BT, and ReGUn preserve substantially more structure than GA/FYE, but their CKA profiles alternate between excessive and insufficient representation changes across depth. ReGUn provides the closest layer-wise baseline, yet still deviates by more than 20% from RE at five layers, with excessive changes in intermediate layers and insufficient changes in deeper layers. In contrast, FF-Erase confines such deviations to only two layers. Overall, the CKA analysis shows why FF unlearning must be evaluated across the complete layer hierarchy: final-layer-only updates leave intermediate representations unchanged, whereas naïve layer-wise extensions can distort representations far beyond retraining. By providing coordinated layer-specific goodness targets, FF-Erase most faithfully approaches the layer-wise representation state induced by retraining while avoiding both failure modes.

I FF VS. BP: OVERALL COMPARISON FROM TRAINING TO UNLEARNING

To assess the practical implications of FF unlearning, we compare complete training–unlearning pipelines against that of BP models in this section. This perspective is particularly relevant to resource-constrained settings, where both training and subsequent unlearning must satisfy runtime and memory budgets. We use the VGG-13 backbone for both FF and BP training on CIFAR-10. After training, we follow the default 20% IID unlearning using the same forget set for all settings. For BP models, we unlearn using SCRUB and ReGUn; For FF models, we unlearn with FF-Erase . We additionally evaluate a four-GPU pipeline-parallel implementation of FF training and FF-Erase to quantify the benefits of layer-wise execution.

Table 27: FF vs. BP: End-to-End Training and Unlearning. The (p) lines report a 4-GPU-parallelized FF implementation.

Method	**Time**(s)↓		**Peak Memory**(MB)↓		**Test Accuracy**(%)		**Forget Accuracy**(%)	
	Train	Unlearn	Train	Unlearn	Train	Unlearn	Train	Unlearn
BP + ReGUn	$1680.5_{\pm 2.3\%}$	$277.8_{\pm 1.3\%}$	$563.1_{\pm 0.4\%}$	$561.7_{\pm 0.3\%}$	$87.03_{\pm 0.6\%}$	$85.28_{\pm 0.7\%}$	$99.46_{\pm 0.1\%}$	$86.30_{\pm 0.4\%}$
BP + SCRUB		$425.5_{\pm 1.7\%}$		$626.08_{\pm 0.7\%}$		$86.31_{\pm 0.6\%}$		$87.98_{\pm 0.5\%}$
FF + FF-Erase	$1388.7_{\pm 2.0\%}$	$139.7_{\pm 3.6\%}$	$205.80_{\pm 0.3\%}$	$238.11_{\pm 0.3\%}$	$82.17_{\pm 0.7\%}$	$79.48_{\pm 0.7\%}$	$91.18_{\pm 0.5\%}$	$81.95_{\pm 1.0\%}$
FF + FF-Erase (p)	$469.0_{\pm 1.7\%}$	$53.5_{\pm 1.6\%}$	$102.05_{\pm 0.4\%}$	$122.38_{\pm 0.5\%}$	$82.19_{\pm 0.7\%}$	$79.44_{\pm 0.7\%}$	$91.17_{\pm 0.6\%}$	$81.79_{\pm 1.2\%}$

Table 27 summarizes the comparison. For the parallel configuration, the reported runtime is the completion time of the slowest GPU, while the memory entry is the maximum peak memory usage among the four GPUs. Per-GPU measurements are provided in Table 28. These memory measurements characterize the evaluated GPU implementation rather than the total system-memory requirements of an edge device.

Table 28: Per-GPU measurements for the four-GPU pipeline-parallel FF implementation.

Metric	**GPU 1**	**GPU 2**	**GPU 3**	**GPU 4**
Training time (s)	$469.0_{\pm 1.7\%}$	$437.8_{\pm 1.2\%}$	$434.6_{\pm 1.8\%}$	$439.5_{\pm 1.5\%}$
Unlearning time (s)	$53.5_{\pm 1.6\%}$	$50.3_{\pm 1.3\%}$	$50.6_{\pm 1.2\%}$	$49.8_{\pm 1.1\%}$
Training peak memory (MB)	$102.05_{\pm 0.4\%}$	$101.33_{\pm 0.3\%}$	$101.35_{\pm 0.3\%}$	$101.06_{\pm 0.3\%}$
Unlearning peak memory (MB)	$122.38_{\pm 0.5\%}$	$121.07_{\pm 0.4\%}$	$121.10_{\pm 0.5\%}$	$120.97_{\pm 0.5\%}$

Interpreting the BP-oriented baselines. SOTA unlearning methods (SCRUB and ReGUn) have demonstrated strong unlearning performance for BP-trained models (Kurmanji et al., 2023; Mirlach et al., 2026). The weaker results of their FF adaptations should therefore be interpreted in the context of transferring globally coupled BP objectives to independently optimized FF layers, rather than as evidence that these methods are generally ineffective. Our findings instead motivate unlearning mechanisms that explicitly account for the layer-local optimization structure of FF models.

Training efficiency and memory requirements. Under the evaluated configurations, single-GPU FF training requires 1388.7 s, compared with 1680.5 s for BP, yielding a $1.21\times$ speedup. The four-GPU FF implementation further reduces training time to 469.0 s, a $2.96\times$ speedup over single-GPU FF. This improvement exploits layer-wise updates that support pipelined execution without an end-to-end backward dependency. The FF pipeline also has a substantially smaller measured memory footprint: its single-GPU peaks are 205.80 MB during training and 238.11 MB during unlearning. Both fall below illustrative 256 MB and 512 MB working-memory budgets, whereas the evaluated BP configurations exceed 512 MB. With four-GPU partitioning, the maximum per-device peaks decrease to 102.05 MB and 122.38 MB, respectively, placing both measured footprints below 128 MB per device. These comparisons concern the reported GPU-memory metric, not total device RAM or a demonstrated deployment on memory-limited edge hardware.

Parallel unlearning and deployment implications. FF-Erase similarly benefits from layer-wise execution: four-GPU parallelism reduces unlearning time from 139.7 s to 53.5 s, yielding a $2.61\times$ speedup over its single-GPU implementation. Although BP remains the predominant training paradigm, these measurements highlight the combination of FF and FF-Erase as a promising option for edge and other resource-constrained deployments where per-device memory capacity and training–unlearning latency are primary constraints.

J LIMITATIONS

FF-Erase is an approximate machine unlearning method and does not provide a formal or certified deletion guarantee. Our evaluation therefore assesses empirical proximity to retraining through forgetting-set behavior, membership inference, retained-data utility, and layer-wise representation similarity. Although FF-Erase remains effective across forgetting ratios from 5% to 50%, the utility gap relative to retraining increases for very large deletion fractions, indicating that extreme removal requests remain more challenging.