A FEW BITS BREAK FORGETTING: BIT-FLIP RELEARNING ATTACK AGAINST MACHINE UNLEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Machine unlearning (MU) is an emerging paradigm to remove specified knowledge from trained models. Despite significant progress, recent studies reveal that unlearned knowledge can be recovered through relearning attacks, raising concerns about MU's robustness. However, existing relearning attacks typically rely on fine-tuning and model update capability for dense parameter modifications, making them detectable by deployment-time integrity verification and less practical. This raises a critical question: can relearning attacks be achieved through only minimal runtime bit-level modifications? As bit-level updates are sparse and discrete, two challenges arise: balancing relearning with utility and efficiently identifying effective bit flips. Existing bit-flip attacks mainly degrade model behavior, which fundamentally differs from relearning and fails to address the challenges. In this paper, we propose **BFRA**, a **B**it-**F**lip **R**elearning **A**ttack that is applicable across tasks and MU methods, enabling fine-tuning-free relearning through minimal bit flips. To balance relearning and utility, BFRA formulates relearning as a constrained discrete optimization problem that maximizes forgotten knowledge recovery while constraining retain utility drop. To improve efficiency, BFRA adopts a progressive bit-flip strategy combining First-Order Candidate Selection and Trial-Loss Bit Picking. Extensive experiments across 3 tasks, 14 model-dataset settings, and 13 MU methods show that BFRA substantially recovers forgotten knowledge with only a few dozen bit flips while keeping high retain utility.

1 INTRODUCTION

Machine unlearning (MU) enables efficient removal of specific targets from trained models (Bourtoule et al., 2021), including training samples, classes or concepts across image classification, image generation, and text generation. MU emerged in response to the General Data Protection Regulation (GDPR) (Voigt & Von dem Bussche, 2017), which grants users the right to withdraw personal data, thus reducing the risk of privacy leakage. MU also serves security and reliability purposes by unlearning harmful or copyrighted data. MU shares a common goal to make the model behave as if the target knowledge had never been learned, while preserving utility on retain knowledge.

Although MU has achieved promising results, recent studies reveal its vulnerability to *relearning attacks*: 'already-forgotten' knowledge can be quickly recovered through fine-tuning with negligible degradation in retain utility. Existing works explore relearning attacks from different perspectives: the type of relearning data, including forget data and auxiliary data (Hu et al., 2025; Deeb & Roger, 2024); the scale of model modification, from full-model to partial parameter updates (Ha et al., 2025); and the target model families, including Large Language Models (LLMs) (Xu et al., 2025; Fan et al., 2025; Lynch et al., 2024), DNNs (Ha et al., 2025; Siddiqui et al., 2026), and diffusion models (Suriyakumar et al., 2024; Liu & Zhang, 2025). These findings suggest that relearning attacks are a practical threat and a valuable perspective for evaluating MU robustness.

However, existing relearning attacks require fine-tuning or model update capabilities, limiting their practicality. In realistic scenarios, such as edge AI systems, attackers may only analyze the model but cannot directly update model files due to deployment-time weight integrity and signature verification. Moreover, the resulting large-scale parameter changes leave a detectable footprint in approx-

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.

imate weight integrity monitoring. Therefore, it raises a critical question ***(Q)****: can relearning attacks be achieved through only minimal runtime bit-level modifications without relying on fine-tuning?*

Bit-flip attacks (BFAs) offer a potential direction: unlike fine-tuning with dense and coordinated updates, BFAs directly modify the binary representation of model weights by flipping a few bits, which can be realized at runtime via Rowhammer-style fault injection without modifying the model file (Kim et al., 2014). However, existing BFAs are mainly designed to degrade model performance, such as causing misclassification (Rakin et al., 2019), implanting backdoors (Chen et al., 2021), or breaking safety alignment (Coalson et al., 2024), which fundamentally differs from relearning attacks. Thus, addressing **(Q)** faces two key challenges: (i) *Relearning-Utility Balance*: sparse and discrete bit flipping provides a highly constrained update space, making it hard to find an optimal relearning solution; meanwhile, directly applying existing BFAs may cause insufficient recovery or utility degradation due to their disruption-oriented objectives. (ii) *Efficiency*: directly searching the effective bit flips from the enormous bit space of large-scale models requires costly exploration.

In this paper, we propose **BFRA**, a **B**it-**F**lip **R**elearning **A**ttack against MU, which can be practically achieved in realistic deployment scenarios with minimal weight modification. Without relying on post-unlearning fine-tuning, BFRA achieves effective relearning attack through only a few dozen bit flips, and is broadly applicable across tasks and unlearning methods. To balance relearning and utility, BFRA adopts a constrained discrete optimization formulation, where the recovery of forgotten knowledge is maximized under the constraint of preserving retain utility. To improve efficiency, we adopt a progressive bit-flip process, where each iteration consists of two stages: First-Order Candidate Selection narrows the search space through approximation, and Trial-Loss Bit Picking identifies effective bit flips through precise loss evaluation. Our contributions are summarized as follows.

- We propose the first bit-flip relearning attack that can substantially recover forgotten knowledge with only dozens of bit flips, revealing a new vulnerability in MU's robustness.
- We formulate bit-flip relearning attacks as a constrained discrete optimization problem, enabling effective recovery of forgotten knowledge while preserving retain utility.
- We propose a progressive bit-flip strategy, where each iteration narrows the search space via approximation and performs precise bit picking, ensuring efficiency in large models.
- We evaluate BFRA against 13 MU methods across 3 tasks and 14 model-dataset settings, demonstrating its broad applicability and superior performance by achieving up to 100% relearning and 100% utility retention with only 25 bit flips.

2 Related Work

2.1 Machine Unlearning

Early MU research mainly focuses on image classification, with typical settings including sample-wise and class-wise MU. Due to the high cost of exact MU, most studies explore approximate MU, such as methods based on parameter manipulation (Foster et al., 2024; Liu et al., 2024), second-order updates (Golatkar et al., 2020; Guo et al., 2020), and gradient-based updates (Chen et al., 2023; Chundawat et al., 2023; Kurmanji et al., 2023). Later, MU isfurther extended to generative models. For image generation, MU is mainly studied as concept or data erasure, with methods including gradient-based updates (Zhang et al., 2024a; Gandikota et al., 2023; Wu et al., 2025), parameter manipulation (Fan et al., 2024; Chavhan et al., 2024), closed-form editing (Gandikota et al., 2024; Gong et al., 2024). For text generation, especially LLMs, the focus shifts to removing textual knowledge. Most methods adopt gradient-based updates with different objectives, including gradient ascent (Yao et al., 2024; Jang et al., 2023), preference optimization (Rafailov et al., 2023; Zhang et al., 2024b; Fan et al., 2026), representation steering (Li et al., 2024).

These studies mainly focus on effective unlearning and output-level evaluation, while our work studies the robustness of unlearned models against bit-level reactivation of forgotten knowledge.

2.2 Bit-Flip Attacks

Bit-Flip Attacks study bit-level vulnerabilities in neural networks, where flipping only a few weight bits can significantly change model behavior. Early studies show that BFAs severely degrade ac-

curacy in image classification (Rakin et al., 2019; Yao et al., 2020). Later works extend BFA to targeted misclassification (Rakin et al., 2021) and Trojan attacks (Rakin et al., 2020; Chen et al., 2021). Recent studies reveal similar vulnerabilities in LLMs, causing performance degradation (Almalky et al., 2025; Guo et al., 2025), safety alignment failure or jailbreaking (Coalson et al., 2024; Zahran et al., 2025; Xu et al., 2026; Das et al., 2025; Guo et al., 2026), and abnormal agent behaviors (Wang et al., 2026). For image generation, studies have shown that bit-level faults can affect generation quality or semantics (Wen et al., 2026; Mubarik et al., 2026; Coalson et al., 2026).

The practicality of BFA is supported by Rowhammer-style hardware attacks, which can induce bit flips in real memory systems on both CPUs (Kim et al., 2014; Kwong et al., 2020; Jattke et al., 2022; 2024; Hassan et al., 2021) and GPUs (Lin et al., 2025; Hu et al., 2026; Wan et al., 2026).

Unlike existing BFAs that use bit flips as fault injections to disrupt model behavior, our goal is to use bit flips for knowledge recovery. This fundamentally different objective introduces problems of insufficient recovery or utility degradation, thus requiring a specific relearning-aware design.

3 Problem Setup

Machine Unlearning. Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ be a training dataset, with the forget set $\mathcal{D}_f \subset \mathcal{D}$ and retain set $\mathcal{D}_r = \mathcal{D} \setminus \mathcal{D}_f$. Let θ_{ori} be the original model trained on $\mathcal{D}$. A MU algorithm $\mathcal{U}$ produces an unlearned model θ_{unl} which should approximate a model θ_{retr} retrained from scratch using only $\mathcal{D}_r$. MU aims to remove the knowledge of $\mathcal{D}_f$ while preserving utility on $\mathcal{D}_r$. For image classification, each example (x_i, y_i) consists of an input image x_i and a class label y_i, and $\mathcal{D}_f$ is constructed from target-class or randomly selected samples. For image generation, each example consists of an image x_i and a target conditioning signal y_i. For text generation, each example consists of an input prompt x_i and a target token sequence $y_i = (y_i^1, \ldots, y_i^{|y_i|})$, where x_i can be empty for standalone passages.

Relearning Attacks. Let $\mathcal{D}_f^{\text{rel}}$ and $\mathcal{D}_r^{\text{rel}}$ be the attacker's relearning data for forgotten and retained knowledge. They can be the original forget/retain data, subsets of them, or data with similar distributions. A relearning algorithm modifies θ_{unl} to produce a relearned model θ_{rel} to recover the behavior of the original model θ_{ori} on the forgotten data, while keeping its behavior on retain data. Thus, it has two goals: (i) *relearning effectiveness*, measures how much forgotten knowledge is recovered; (ii) *retain utility* measures how well the model's utility on retain data is preserved after the attack.

Threat Model. We consider a white-box adversary $\mathcal{A}$, consistent with previous BFA work (Xu et al., 2026). $\mathcal{A}$ can download the unlearned model θ_{unl}, including its architecture and parameters. It can perform offline analysis through forward and backward propagations using a forget-related relearning set $\mathcal{D}_f^{\text{rel}}$ and a retain-related set $\mathcal{D}_r^{\text{rel}}$. However, $\mathcal{A}$ is not allowed to directly replace the deployed model by uploading a modified model back to the service provider. Instead, after identifying vulnerable bits offline, $\mathcal{A}$ is able to use Rowhammer to flip these bits at runtime without modifying the model file. Edge AI systems are a realistic example, which mainly rely on deployment-time integrity verification(e.g., model signature and hash), while runtime verification of in-memory weight integrity remains challenging due to overhead Yang et al. (2025). Thus, unauthorized model updates can be easily detected, while Rowhammer-induced BFAs may evade.

4 Method

To ensure relearning-utility balance, we formulate relearning as a *constrained discrete optimization problem*, where the objective is to minimize the training loss $\mathcal{L}$ on the forget-related relearning dataset $\mathcal{D}_f^{\text{rel}}$ while constraining the degradation of the training loss on the retain-related relearning dataset $\mathcal{D}_r^{\text{rel}}$. The specific form of $\mathcal{L}$ depends on the model, e.g., cross-entropy loss for classification models, denoising loss for diffusion models, and token-level cross-entropy for LLMs. In practice, the required losses and gradients can be computed on mini-batches instead of full datasets.

Our BFRA adopts a progressive strategy, where one bit is selected and permanently flipped in each iteration. Specifically, let T denote the maximum number of iterations, and t the current iteration index. We denote the model after t bit flips as θ^t, with the initial model $\theta^0 = \theta_{\text{unl}}$. This process continues until T is reached or no qualified candidate remains. Compared with simultaneously flipping multiple bits, this enables fine-grained control of the recovery-utility trade-off. In each

iteration, BFRA performs two stages to balance efficiency and accuracy: (i) First-Order Candidate Selection uses gradient-based loss approximation to efficiently identify promising bit flips; (ii) Trial-Loss Bit Picking further picks the optimal candidate through actual loss computation.

For efficiency, we restrict bit flips to the sign and exponent bits. Mantissa bit flips usually induce much smaller weight perturbations and thus have limited impact on model behavior. For an FP32 weight, this reduces the bit search space from 32 to 9 bits, eliminating over 70% of bit candidates.

4.1 Stage 1: First-Order Candidate Selection

Since evaluating the exact loss change of every bit flip is computationally expensive, we use first-order approximation to narrow the search space. Specifically, we filter out bit flips that may largely affect the retain utility, and then select candidates with high estimated relearning potential. To further improve efficiency, we introduce a weight-level skip search strategy.

Let w_i denote the i-th weight, and let $b_{i,k}$ denote flipping the k-th bit of w_i. Flipping $b_{i,k}$ only changes w_i and induces a weight perturbation $\Delta w_{i,k}$. Using the first-order Taylor approximation, the loss changes on the retain-related set $\mathcal{D}_r^{\text{rel}}$ and the forget-related set $\mathcal{D}_f^{\text{rel}}$ can be estimated as

$$\begin{aligned}\Delta\mathcal{L}_{i,k}^r &= \mathcal{L}(\theta + \Delta w_{i,k}\mathbf{e}_i; \mathcal{D}_r^{\text{rel}}) - \mathcal{L}(\theta; \mathcal{D}_r^{\text{rel}}) \approx g_i^r \Delta w_{i,k}, \\ \Delta\mathcal{L}_{i,k}^f &= \mathcal{L}(\theta + \Delta w_{i,k}\mathbf{e}_i; \mathcal{D}_f^{\text{rel}}) - \mathcal{L}(\theta; \mathcal{D}_f^{\text{rel}}) \approx g_i^f \Delta w_{i,k},\end{aligned} \tag{1}$$

where $\mathbf{e}_i$ is the unit vector corresponding to weight w_i, and $g_i^r = \frac{\partial\mathcal{L}(\theta;\mathcal{D}_r^{\text{rel}})}{\partial w_i}$ and $g_i^f = \frac{\partial\mathcal{L}(\theta;\mathcal{D}_f^{\text{rel}})}{\partial w_i}$ are the gradients with respect to w_i on $\mathcal{D}_r^{\text{rel}}$ and $\mathcal{D}_f^{\text{rel}}$, respectively.

Based on these, among the eligible bits set $\mathcal{B}$, our goal is to construct a candidate pool $\mathcal{P}$ that stores the top-K candidate bit flips with the largest estimated loss reduction on the forget-related set $\mathcal{D}_f^{\text{rel}}$, while constraining their estimated impact on the retain-related set $\mathcal{D}_r^{\text{rel}}$ within a budget ϵ_r:

$$\begin{aligned}\mathcal{P}^* = \arg\max_{\mathcal{P}\subseteq\mathcal{B}} &\sum_{b_{i,k}\in\mathcal{P}} \left(-g_i^f \Delta w_{i,k}\right) \\ \text{s.t.} \quad &|\mathcal{P}| = K, \qquad |g_i^r \Delta w_{i,k}| \le \epsilon_r, \quad \forall b_{i,k} \in \mathcal{P}.\end{aligned} \tag{2}$$

Here, the negative sign in $(-g_i^f \Delta w_{i,k})$ makes larger values correspond to larger estimated $\mathcal{D}_f^{\text{rel}}$ loss reduction. We use $|g_i^r \Delta w_{i,k}|$ to constrain the magnitude of the estimated retain-loss change, since our goal is to preserve the unlearned model's existing performance on $\mathcal{D}_r^{\text{rel}}$.

Skip Search. Directly solving the constrained Top-K selection requires exhaustive bit flipping, which is costly for large models. Thus, we propose skip search to efficiently identify candidate bits.

Based on the retain constraint, each weight change Δw_i has an allowable range. To avoid division by zero or overly large ranges when $|g_i^r|$ is very small, we add a small stabilizing constant ρ:

$$\Delta w_i \in [\Delta w_i^{\min}, \Delta w_i^{\max}], \quad \Delta w_i^{\min} = -\frac{\epsilon_r}{|g_i^r| + \rho}, \quad \Delta w_i^{\max} = \frac{\epsilon_r}{|g_i^r| + \rho}. \tag{3}$$

Since all bit flips within weight w_i share the same gradients g_i^r and g_i^f , we compute a shared weight-level upper bound of the estimated $\mathcal{D}_f^{\text{rel}}$ loss reduction over the allowable range of Δw_i:

$$U_i^f = \max_{\Delta w\in[\Delta w_i^{\min}, \Delta w_i^{\max}]} \left(-g_i^f \Delta w\right) = \begin{cases} -g_i^f \Delta w_i^{\min}, & g_i^f \ge 0, \\ -g_i^f \Delta w_i^{\max}, & g_i^f < 0. \end{cases} \tag{4}$$

The weights in each layer are sorted according to U_i^f in descending order. Among the current candidate pool $\mathcal{P}$, we maintain the minimum estimated $\mathcal{D}_f^{\text{rel}}$ loss reduction:$\tau^f = \min_{b_{j,k}\in\mathcal{P}}(-g_j^f \Delta w_{j,k})$.

When $\mathcal{P}$ is not full, we sequentially traverse the sorted weights and flipping each bit $b_{i,k}$ to compute $-g_i^f \Delta w_{i,k}$ and $|g_i^r \Delta w_{i,k}|$. Qualified bit flips are inserted into $\mathcal{P}$, and τ^f is updated accordingly.

Once $\mathcal{P}$ is full, if the upper bound of weight w_i satisfies $U_i^f \le \tau^f$, all remaining weights in the current layer can be skipped. Otherwise, bit flips in w_i are evaluated and inserted into $\mathcal{P}$ if qualified.

Essentially, our skip strategy performs weight-level pruning, avoiding unnecessary first-order bit-flip evaluations for most weights while maintaining the quality of the candidate pool.

Moreover, constraining Δw_i improves the reliability of first-order estimation. Large perturbations caused by exponent or sign bit flips may violate the local perturbation assumption of Taylor approximation, leading to inaccurate loss estimation. Extreme updates also cause numerical instability, such as activation explosion or NaN values, completely damaging the relearning process.

Layer Quota. Furthermore, due to model's structural characteristics, some layers may naturally produce larger gradients, causing their bits to dominate the candidate pool. Thus, we introduce a dynamic layer quota mechanism to balance candidate selection across layers. The quota is determined by layer size and historical contribution during progressive flipping. For each layer l, we compute a smoothed historical hit rate: $r_l = \frac{n_{\text{final}}+1}{n_{\text{candidate}}+2}$, where $n_{\text{candidate}}$ and n_{final} denote the numbers of candidate and finally selected bits from layer l, respectively. Then, the layer quota is

$$K_l = K \cdot \frac{n_l r_l}{\sum_j n_j r_j}, \tag{5}$$

where n_l denotes the number of bits in layer l. Thus, larger layers receive larger initial quotas, while layers that historically produced more successful flips progressively gain larger quotas.

4.2 Stage 2: Trial-Loss Bit Picking

Although the first-order candidate selection efficiently identifies promising bit flips, the estimated relearning score may deviate from the actual loss change due to the approximation error of Taylor expansion. Therefore, directly selecting the highest-scored bit may lead to suboptimal performance.

Given the candidate pool $\mathcal{P}$, we temporarily flip each candidate bit $b_{i,k}$ and evaluate its actual impact on the losses over $\mathcal{D}_f^{\text{rel}}$ and $\mathcal{D}_r^{\text{rel}}$. Here, $\Delta w_{i,k}$ denotes the scalar perturbation induced by flipping the k-th bit of weight w_i, and $\mathbf{e}_i$ is the unit vector corresponding to w_i. We pick the bit with the largest loss reduction on $\mathcal{D}_f^{\text{rel}}$ under two retain loss constraints:

$$\begin{aligned} b^* = \arg\max_{b_{i,k}\in\mathcal{P}} \quad & -\left[\mathcal{L}(\theta^t + \Delta w_{i,k}\mathbf{e}_i; \mathcal{D}_f^{\text{rel}}) - \mathcal{L}(\theta^t; \mathcal{D}_f^{\text{rel}})\right] \\ \text{s.t.} \quad & \mathcal{L}(\theta^t + \Delta w_{i,k}\mathbf{e}_i; \mathcal{D}_r^{\text{rel}}) - \mathcal{L}(\theta^t; \mathcal{D}_r^{\text{rel}}) \le \epsilon_s, \\ & \mathcal{L}(\theta^t + \Delta w_{i,k}\mathbf{e}_i; \mathcal{D}_r^{\text{rel}}) - \mathcal{L}(\theta^0; \mathcal{D}_r^{\text{rel}}) \le \epsilon_c. \end{aligned} \tag{6}$$

The first constraint controls the retain loss caused by each bit flip, and the second controls the total retain loss across progressive flipping. Here, ϵ_s and ϵ_c are the step-wise and cumulative budgets.

5 Experiments

5.1 Experimental Settings

The experiments are conducted in Python 3.9 with 8 NVIDIA GeForce RTX 4090. For image classification, we use CIFAR-10 and CIFAR-100 (Krizhevsky et al., 2009), on AllCNN (Springenberg et al., 2014), ResNet-18 (He et al., 2016), and ViT (Dosovitskiy, 2020). For image generation, we evaluate class forgetting on CIFAR-10 with DDPM (Ho et al., 2020). For LLMs, we evaluate BFRA on TOFU (Maini et al., 2024) with Llama-3.2-1B (Grattafiori et al., 2024), Llama-2-7B (Touvron et al., 2023) and Phi-1.5 (Li et al., 2023), on WMDP (Li et al., 2024) with Zephyr-7B-BETA (Tunstall et al., 2023). See Appendix B for detailed settings and Appendix C for extended experiments.

Baselines. For image classification, we consider 3 retain-free MU methods: gradient ascent (GA), boundary shrink (BS) (Chen et al., 2023), Saliency Unlearning (SalUn) (Fan et al., 2024); and 2 retain-based: fine-tuning (FT) and SCRUB (Kurmanji et al., 2023). For image generation, we consider: Random Labeling (RL), SalUn, Erased Stable Diffusion (ESD) (Gandikota et al., 2023), and Forget-Me-Not (FMN) (Zhang et al., 2024a). For LLMs, we consider: GA, Negative Preference Optimization (NPO) (Zhang et al., 2024b), Representation Misdirection Unlearning (RMU) (Li et al., 2024), and simNPO (Fan et al., 2026). For BFA comparison, we consider: BFA (Rakin et al., 2019), SBFA (Guo et al., 2025), SilentStriker (Xu et al., 2026) and AttnBreaker (Das et al., 2025).

Table 1: Class unlearning on image classifiers, CIFAR-10. † indicates BFRA-relearned models.

Model	AllCNN, CIFAR-10				ResNet-18, CIFAR-10				ViT, CIFAR-10			
Metric	FA	RA	TA	T	FA	RA	TA	T	FA	RA	TA	T
Original	$99.48_{\pm 0.00}$	$99.73_{\pm 0.00}$	$84.20_{\pm 0.00}$	-	$99.96_{\pm 0.00}$	$99.84_{\pm 0.00}$	$92.50_{\pm 0.00}$	-	$94.66_{\pm 0.00}$	$92.52_{\pm 0.00}$	$79.37_{\pm 0.00}$	-
Retrain	$0.00_{\pm 0.00}$	$99.96_{\pm 0.00}$	$84.29_{\pm 0.00}$	-	$0.00_{\pm 0.00}$	$99.86_{\pm 0.00}$	$92.68_{\pm 0.08}$	-	$0.00_{\pm 0.00}$	$93.16_{\pm 0.00}$	$80.28_{\pm 0.00}$	-
GA	$0.00_{\pm 0.00}$	$88.21_{\pm 0.12}$	$73.47_{\pm 0.19}$	-	$3.04_{\pm 1.08}$	$85.35_{\pm 1.14}$	$79.50_{\pm 1.38}$	-	$2.46_{\pm 0.89}$	$76.70_{\pm 1.31}$	$69.17_{\pm 1.55}$	-
GA†	$100.0_{\pm 0.00}$	$93.76_{\pm 0.27}$	$77.28_{\pm 0.49}$	28	$99.10_{\pm 0.14}$	$85.03_{\pm 0.32}$	$78.34_{\pm 0.52}$	39	$99.96_{\pm 0.04}$	$73.45_{\pm 1.05}$	$65.06_{\pm 1.24}$	58
BS	$0.84_{\pm 0.19}$	$91.18_{\pm 0.81}$	$74.76_{\pm 1.02}$	-	$0.36_{\pm 0.14}$	$85.83_{\pm 0.71}$	$78.50_{\pm 0.95}$	-	$1.08_{\pm 0.21}$	$83.03_{\pm 0.58}$	$73.46_{\pm 0.72}$	-
BS†	$99.98_{\pm 0.02}$	$92.45_{\pm 0.45}$	$74.48_{\pm 0.66}$	43	$100.0_{\pm 0.00}$	$88.02_{\pm 0.24}$	$80.28_{\pm 0.40}$	40	$99.56_{\pm 0.10}$	$75.94_{\pm 0.96}$	$66.84_{\pm 1.15}$	65
SalUn	$1.74_{\pm 0.86}$	$85.04_{\pm 1.12}$	$70.22_{\pm 1.34}$	-	$0.24_{\pm 0.09}$	$97.78_{\pm 0.28}$	$89.77_{\pm 0.41}$	-	$3.70_{\pm 1.25}$	$77.32_{\pm 1.82}$	$69.58_{\pm 2.20}$	-
SalUn†	$99.98_{\pm 0.02}$	$93.24_{\pm 0.22}$	$69.13_{\pm 0.38}$	27	$100.0_{\pm 0.00}$	$97.02_{\pm 0.32}$	$88.19_{\pm 0.49}$	8	$100.0_{\pm 0.00}$	$70.44_{\pm 0.30}$	$61.84_{\pm 0.34}$	74
FT	$0.00_{\pm 0.00}$	$97.77_{\pm 0.17}$	$82.56_{\pm 0.33}$	-	$0.00_{\pm 0.00}$	$96.24_{\pm 0.42}$	$87.46_{\pm 0.60}$	-	$0.00_{\pm 0.00}$	$87.87_{\pm 0.52}$	$58.35_{\pm 0.60}$	-
FT†	$99.30_{\pm 0.63}$	$88.80_{\pm 0.38}$	$74.38_{\pm 0.51}$	49	$98.02_{\pm 0.39}$	$87.02_{\pm 1.03}$	$78.19_{\pm 1.17}$	53	$81.76_{\pm 1.42}$	$87.16_{\pm 2.11}$	$75.53_{\pm 2.68}$	13
SCRUB	$0.00_{\pm 0.00}$	$99.38_{\pm 0.08}$	$83.63_{\pm 0.13}$	-	$0.00_{\pm 0.00}$	$99.11_{\pm 0.12}$	$89.18_{\pm 0.22}$	-	$0.02_{\pm 0.02}$	$91.68_{\pm 0.11}$	$79.90_{\pm 0.12}$	-
SCRUB†	$90.22_{\pm 0.34}$	$94.28_{\pm 0.42}$	$78.16_{\pm 0.65}$	22	$94.58_{\pm 0.12}$	$89.35_{\pm 0.35}$	$79.27_{\pm 0.48}$	47	$99.02_{\pm 0.30}$	$82.11_{\pm 0.66}$	$70.88_{\pm 0.49}$	49

Table 2: Class unlearning results on DDPM, CIFAR-10. † indicates BFRA-relearned results.

Class	airplane					automobile					cat					dog					ship				
Metrics	FA	FID_f	RA	FID_r	T	FA	FID_f	RA	FID_r	T	FA	FID_f	RA	FID_r	T	FA	FID_f	RA	FID_r	T	FA	FID_f	RA	FID_r	T
original	92.20	97.04	98.13	26.11	-	100.0	56.26	97.27	23.25	-	93.60	99.75	97.98	22.47	-	96.00	89.92	97.71	22.31	-	99.40	66.12	97.33	22.72	-
RL	2.41	165.89	97.36	29.29	-	0.00	208.60	97.73	20.20	-	1.72	191.84	97.03	23.69	-	1.36	210.19	96.31	23.02	-	0.41	167.58	97.54	22.45	-
RL†	79.39	106.32	96.89	28.66	10	95.20	80.50	95.44	24.04	30	83.60	113.14	93.05	21.07	30	76.64	107.67	94.38	20.76	90	96.43	64.92	92.31	25.20	70
SalUn	2.03	174.12	97.41	27.95	-	0.00	212.71	96.56	27.82	-	2.22	174.02	97.24	23.46	-	2.96	202.54	96.64	22.56	-	0.41	150.31	97.71	21.97	-
SalUn†	88.39	99.84	93.28	39.12	40	87.80	70.23	87.04	25.99	80	88.02	123.14	96.08	21.68	30	83.19	105.82	83.53	28.29	30	95.60	67.87	94.89	22.68	50
ESD	1.84	200.31	89.93	44.11	-	0.62	219.72	95.76	29.09	-	2.60	170.33	94.04	31.43	-	1.22	182.39	94.02	31.17	-	0.00	266.42	94.04	32.83	-
ESD†	93.00	97.65	89.64	48.15	90	99.02	92.04	94.31	30.69	30	88.84	127.61	88.42	31.89	60	90.01	103.26	93.57	28.50	90	99.62	57.84	93.23	30.92	30
FMN	1.21	164.04	97.53	28.03	-	0.00	210.96	97.47	26.33	-	0.00	212.66	96.60	27.74	-	0.00	218.95	96.29	30.43	-	0.80	168.32	96.56	27.82	-
FMN†	92.82	93.13	96.50	29.51	20	99.19	59.50	96.80	23.52	40	93.04	101.99	97.73	20.94	10	84.42	109.93	97.49	23.29	20	99.23	60.60	86.08	27.12	30

Evaluation Metrics. For image classification, we report forgetting accuracy (FA), retain accuracy (RA), and test accuracy (TA) on $\mathcal{D}_f$, $\mathcal{D}_r$, and the test set $\mathcal{D}_t$, respectively. For DDPM, we use an external classifier (ResNet-34) to measure FA and RA, and report FID_f and FID_r for the generation quality on $\mathcal{D}_f$ and $\mathcal{D}_r$. For LLMs on TOFU, we report the probability (Prob) and ROUGE on forget set, Model Utility (MU), and Forget Quality (FQ). On WMDP, we use multiple-choice accuracy on the target (WMDP-bio and cyber) and retain knowledge (MMLU). Since recovery often trades off utility, we report the Tth bit-flip with the highest FA/Prob while limiting the RA/MU drop to within 10% of the unlearned model. Results in Table 1 are averaged over 5 seeds, including T.

5.2 BFRA in Different Tasks

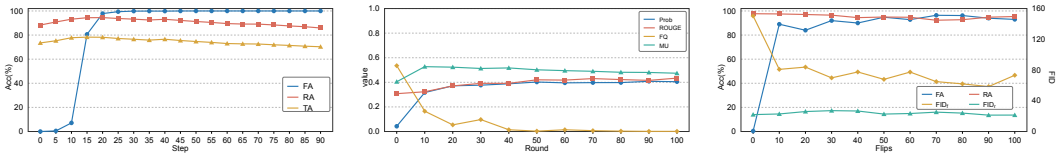
We evaluate BFRA on class-1 unlearning for image classifiers, class unlearning for DDPM, and TOFU-1% unlearning for LLMs. Tables 1, 2 and 3 show that BFRA substantially recovers forgotten knowledge while preserving utility across settings, revealing broad bit-level relearning vulnerability.

Table 3: LLM unlearning results on TOFU-Forget1%. † indicates BFRA-relearned results.

Model	Llama3.2-1B					Llama2-7B					Phi-1.5B				
Metric	Prob	ROUGE	MU	FQ	T	Prob	ROUGE	MU	FQ	T	Prob	ROUGE	MU	FQ	T
original	0.9011	0.8731	0.6033	0.0100	-	0.9951	0.9493	0.6276	1.3e-3	-	0.9233	0.8424	0.3910	0.0005	-
retrain	0.1655	0.4071	0.6054	1.0000	-	0.1845	0.4126	0.6282	1.0000	-	0.1503	0.3877	0.4082	1.0000	-
GA	0.0427	0.3077	0.5352	0.4046	-	0.0064	0.1767	0.5317	0.0541	-	0.0768	0.4165	0.3660	0.7659	-
GA†	0.4024	0.4205	0.5019	0.0030	50	0.6761	0.4539	0.6087	0.0013	20	0.3572	0.4607	0.3421	0.2786	100
NPO	0.0676	0.2743	0.5849	0.9188	-	0.0684	0.3043	0.5759	0.9900	-	0.1091	0.4185	0.3813	0.7659	-
NPO†	0.2494	0.3730	0.5284	0.4046	40	0.4243	0.3922	0.5305	0.1650	90	0.3829	0.4728	0.3425	0.0971	100
RMU	0.0561	0.2115	0.5723	0.9188	-	0.4647	0.4053	0.6305	0.2657	-	0.0005	0.0359	0.3965	0.0005	-
RMU†	0.6515	0.5079	0.5349	0.0068	50	0.8647	0.7133	0.6152	0.0068	100	0.6759	0.5214	0.3999	0.0541	100
simNPO	0.1575	0.2677	0.5432	0.9188	-	0.3528	0.4846	0.6170	0.0030	-	0.2674	0.4299	0.3974	0.0286	-
simNPO†	0.9055	0.9046	0.5337	0.0143	20	0.7463	0.5389	0.6067	0.0068	40	0.5896	0.5123	0.3886	0.0013	100

Table 4: Relearning comparison with BFAs and Fine-tuning. † indicates BFRA-relearned results.

Model	AllCNN, CIFAR-10				Llama3.2-1B, TOFU-1%					DDPM, CIFAR10				
Metric	FA	RA	TA	T	Prob	ROUGE	MU	FQ	T	FA	FID_f	RA	FID_r	T
Original	99.48	99.73	84.20	-	0.9011	0.8731	0.6033	0.0100	-	100.0	56.26	97.27	23.25	-
Unlearned	0.00	88.21	73.47	-	0.0427	0.3077	0.5352	0.4046	-	0.00	208.60	97.73	20.20	-
Finetune-1	68.52	94.56	77.78	-	0.2000	0.3987	0.5757	0.2657	-	90.80	59.71	89.51	24.62	-
Finetune-5	99.30	97.38	80.19	-	0.6394	0.5020	0.5583	0.0068	-	90.40	73.58	71.31	37.24	-
BFA	99.98	6.41	5.02	1	0.0427	0.3077	0.5352	0.4046	0	0.00	208.60	97.73	20.20	0
SBFA	100.0	37.55	29.44	45	0.2156	0.3449	0.3306	0.5786	6	0.20	202.48	96.33	22.82	2
SilentStriker	2.74	92.72	76.69	28	0.2993	0.3780	0.4540	0.5786	29	7.20	206.20	96.64	22.31	20
AttnBreaker	100.0	13.21	10.78	60	0.0427	0.3077	0.5352	0.4046	0	60.05	102.66	95.70	22.91	70
BFRA	100.0	93.76	77.28	28	0.4024	0.4205	0.5019	0.0030	50	95.20	80.50	95.44	24.04	30



(a) AllCNN, CIFAR-10, GA (b) Llama3.2-1B, TOFU-1%, GA (c) DDPM, CIFAR-10, RL

Figure 1: Relearning Trajectory.

For small models, BFRA achieves near-full recovery with high utility. In image classifiers, recovered FA often exceeds 99%, and DDPM shows similar trends that BFRA recovers about 90% FA and reduces FID_f to about 100. For LLMs, whose size is orders of magnitude larger, BFRA is weaker but still effective. In particular, models unlearned by simNPO and RMU exhibit strong relearning, with the highest Prob reaching 0.9055. Interestingly, larger LLMs (e.g., Llama2-7B) often show stronger relearning, possibly due to their higher capacity preserving more recoverable knowledge.

5.3 Comparison with Fine-tuning and BFAs

We compare BFRA with fine-tuning and BFA baselines in Table 4. BFRA generally outperforms one-epoch fine-tuning, while five epochs achieve stronger relearning. Note that our goal is not to surpass fine-tuning, but to enable practical relearning without dense updates. Existing BFAs fail to balance recovery and utility, highlighting the benefit of BFRA's relearning-aware design.

5.4 Relearning Trajectory

We plot the metric trajectories throughout BFRA. As shown in Fig. 1, BFRA exhibits rapid early-stage recovery, with substantial relearning from only 20 flips. The marginal benefit of further flips decreases, while retain utility starts to degrade, revealing a clear relearning–utility trade-off.

5.5 Ablation Study

We first conduct a module-level ablation, as shown in Table 5a. On AllCNN, removing the constraints still allows near-full recovery, but severely degrades retain utility. Adding the first-order constraint (FOC) not only preserves utility but, more importantly, restricts the feasible perturbation range, preventing premature termination (within 2 flips) due to activation explosion or NaNs. Combining FOC and trial-loss constraints (TLC) substantially improves retain utility preservation. LQ further improves performance by diversifying the search across layers, with a more pronounced benefit on LLMs. FOC is coupled with SKIP search and is thus critical for search efficiency. Disabling FOC forces exhaustive first-order parameter scanning, making TLC+LQ extremely slow. On AllCNN/CIFAR-10, the runtime increases from 6.01 s/round to 484.8 s/round, an $80.6\times$ slowdown.

Table 5b compares different bit fields. Exponent bits generally provide stronger relearning, while mantissa-based search requires more flips due to weaker perturbations. Mantissa bits remain effec-

Table 5: Ablation study on image classification and text generation task.

(a) Ablation on modules. Here, FOC is short for First-Order Constraint on $\mathcal{D}_r^{\text{rel}}$; TLC is short for Trial-Loss Constraint on $\mathcal{D}_r^{\text{rel}}$; LQ is short for Layer Quota.

Module			AllCNN, CIFAR-10				Llama3.2-1B, TOFU-1%				
FOC	TLC	LQ	FA	RA	TA	T	Prob	ROUGE	MU	FQ	T
Unlearned			0.00	88.21	73.47	-	0.0427	0.3077	0.5352	0.4046	-
-	-	-	99.98	6.41	5.02	1	0.0427	0.3077	0.5352	0.4046	0
✓	-	-	100.00	71.39	60.42	35	0.2968	0.3871	0.4335	0.1649	20
-	✓	-	99.90	9.12	7.29	2	0.0427	0.3077	0.5352	0.4046	0
-	-	✓	100.00	0.04	0.04	1	0.0427	0.3077	0.5352	0.4046	0
✓	✓	-	99.78	92.48	75.89	30	0.2236	0.3451	0.5050	0.5786	2
✓	-	✓	100.00	51.18	43.71	50	0.3088	0.4017	0.4232	0.1650	10
-	✓	✓	99.96	3.18	2.59	17	0.0427	0.3077	0.5352	0.4046	0
✓	✓	✓	100.0	93.76	77.28	28	0.4024	0.4205	0.5019	0.0030	50

(b) Ablation on floating-point bit-field selection. Here, Sign, Exp, and Man denote the sign, exponent, and mantissa bits, respectively.

Bit-Field			AllCNN, CIFAR-10				Llama3.2-1B, TOFU-1%				
Sign	Exp	Man	FA	RA	TA	T	Prob	ROUGE	MU	FQ	T
Original			99.48	99.73	84.20	-	0.9011	0.8731	0.6033	0.0100	-
Unlearned			0.00	88.21	73.47	-	0.0427	0.3077	0.5352	0.4046	-
✓	-	-	99.66	85.84	69.41	70	0.2960	0.3920	0.5122	0.2657	100
-	✓	-	100.0	92.48	75.76	55	0.3814	0.4260	0.4901	0.0541	50
-	-	✓	100.0	94.21	76.57	85	0.2113	0.3682	0.5216	0.4046	100
✓	✓	-	100.0	93.76	77.28	28	0.4024	0.4205	0.5019	0.0030	50
✓	-	✓	99.90	80.37	65.63	100	0.3149	0.3776	0.4946	0.0541	100
-	✓	✓	100.0	92.70	75.80	40	0.3639	0.4142	0.4823	0.0068	100
✓	✓	✓	100.0	92.05	75.23	45	0.3605	0.3942	0.4599	0.0971	90

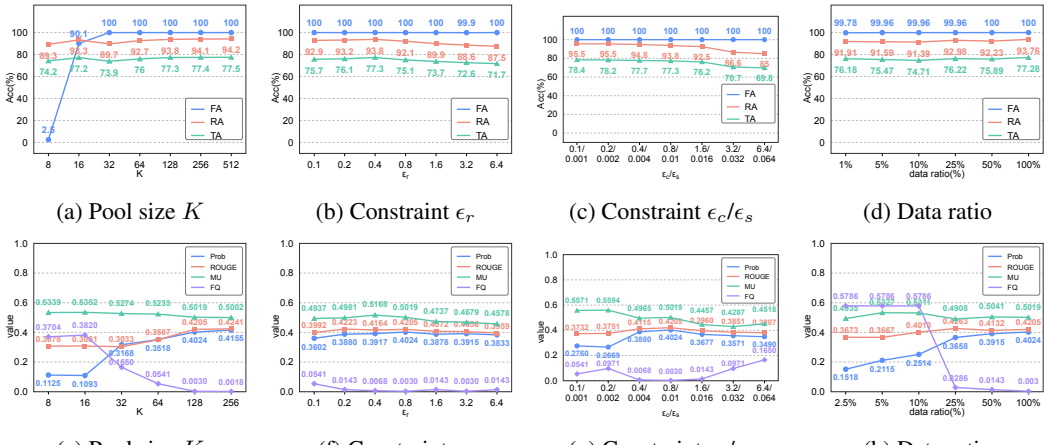
tive for small classifiers but degrade recovery and utility on LLMs. Sign+exponent achieves the best relearning–utility balance with fewer flips. Searching all fields can even underperform and converge slowly, likely because mantissa bits divert the progressive search toward suboptimal trajectories.

5.6 Hyperparameter Analysis

We analyze BFRA's sensitivity to the candidate pool size K, first-order constraint ϵ_r, trial-loss constraints ϵ_c/ϵ_s, and accessible relearning data ratio. As shown in Fig. 2, a small K limits the search space and weakens recovery, while $K = 128$ is sufficient. BFRA is robust to constraint budgets: overly strict constraints slightly reduce recovery while overly loose ones hurt retain utility. BFRA in small image classifiers remain strong with 1% data, whereas LLM relearning degrades noticeably with 10% or lower data, suggesting that textual knowledge requires broader data coverage.

5.7 Efficiency Analysis

In Table 6, we analyze search-space reduction and runtime breakdown averaged over iterations across 3 tasks. Restricting Eligible Bits to sign and exponent bits reduces the search space by 43.75%, 62.5%, and 71.88% for BF16, FP16, and FP32 weights, respectively. SKIP search further avoids up to 99.99% of eligible-bit evaluations, reducing Stage 1 search to about 1s. Stage 2 dominates runtime due to the necessary trial-loss evaluations over 128 candidates. But our additional trial-loss constraints add only limited overhead, as they are applied to less than 1/3 candidates on average. Overall, BFRA ensures efficiency by avoiding exhaustive evaluation over the bit space.



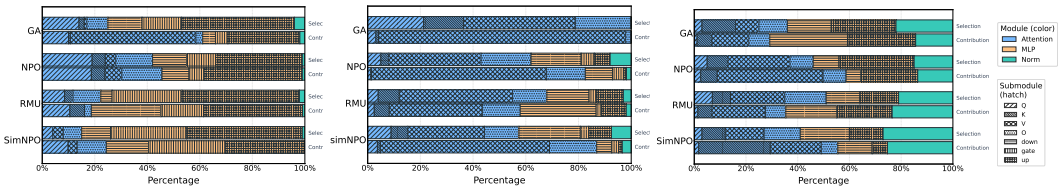
(a) Pool size K (b) Constraint ϵ_r (c) Constraint ϵ_c/ϵ_s (d) Data ratio

(e) Pool size K (f) Constraint ϵ_r (g) Constraint ϵ_c/ϵ_s (h) Data ratio

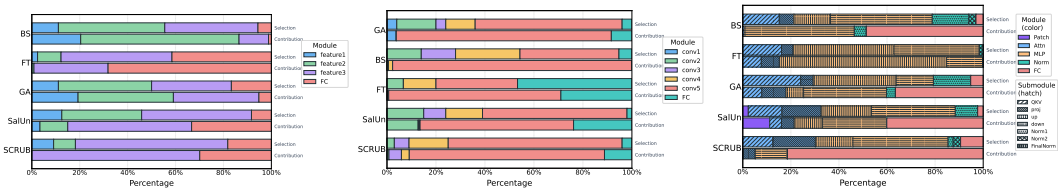
Figure 2: Hyperparameter analysis of BFRA on GA-unlearned model. Figures a–d are on AllCNN and CIFAR-10, while Figures e–h are on Llama3.2-1B, TOFU-1%.

Table 6: Runtime and bit count comparison. 'F+B' indicates the total forward and backward propagation for both $\mathcal{D}_f^{\text{rel}}$ and $\mathcal{D}_r^{\text{rel}}$. 'Cum./Step Rej' indicate the bits rejected by ϵ_c/ϵ_s constraints.

	Stage 1 Time			Stage 2 Time			Bits Count			
Settings	F+B	Search	Total	F+B	Search	Total	Eligible	SKIP	Cum. Rej	Step Rej
ResNet18, CIFAR100	0.28	0.10	0.38	5.26	0.03	5.29	100,937,088	85.87%	9.00	27.54
Llama3.2-1B, TOFU1%	1.14	0.85	2.23	11.84	0.28	12.12	8,758,315,008	95.46%	0.00	4.89
DDPM, CIFAR10	0.14	1.48	1.63	3.20	0.09	3.29	347,689,755	99.99%	0.00	0.84



(a) Llama3.2-1B, TOFU-1% (b) Llama2-7B, TOFU-1% (c) Phi-1.5B, TOFU-1%



(d) AllCNN, CIFAR-100 (e) ResNet18, CIFAR-100 (f) ViT, CIFAR-100

Figure 3: Module-level selection and contribution distribution for LLMs and image classifiers.

5.8 Discussion: Structural Patterns of Relearning Bits

We analyze the selection and contribution distributions of relearning bits in Fig. 3, where contribution is measured by the $\mathcal{D}_f$ loss drop from each flip. Selection does not always match contribution: in Llama3.2-1B with GA, Attention contributes over twice its selection ratio, and one v_{proj} bit contributes 47.8% of recovery. For transformers, Attention and MLP dominate relearning bits. Larger models tend to rely more on Attention bits, while MLP bits remain important in smaller ones. This may be because larger models preserve more redundant representations, making attention-based rerouting sufficient for recovery. Within Attention, V projections often contribute mostly, possibly because it directly controls the propagated information, while Q/K mainly determine where to attend among tokens. Within MLP, effective bits mainly appear in up/down projections, possibly because they directly control feature transformation. For CNNs (Fig. 3d to 3f), relearning bits exhibit clearer layer-level patterns, focusing on middle-to-high layers. This suggests that relearning vulnerability in CNNs is mainly associated with high-level semantic representations and prediction mappings.

We further analyze bit positions and find that the top three selected bit positions typically contribute over 90% of recovery. Since our first-order constraints suppress large perturbations, effective flips tend to concentrate on a few exponent bits, typically bits 8–10 for BF16 and bits 24–26 for FP32.

6 Conclusion

In this paper, we proposed BFRA, a bit-flip relearning attack that recovers forgotten knowledge from unlearned models without relying on fine-tuning. BFRA reveals that even a few sparse bit flips can reactivate forgotten knowledge while largely preserving retain utility. Specifically, BFRA formulates relearning as a constrained discrete optimization problem and adopts a progressive two-stage strategy. Extensive experiments across different tasks and MU methods show BFRA's strong relearning performance with only a small number of bit flips. Our findings reveal a new vulnerability of MU and highlight the need to evaluate MU methods against bit-level relearning attacks.

7 AI USE STATEMENT

In this work, we used generative AI tools for providing feedback on research methodology or experiments, implementing methods, assisting with translation. We have not used generative AI tools for designing research methodology or experiments, cleaning and reformatting dataset, support qualitative and thematic data analysis. Generating synthetic data sets, helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refining hypotheses are not applicable to this work. Additionally, we used generative AI tools for creating or editing software code, sourcing/searching for information, editing a research paper to improve readability, identifying relevant literature, suggesting a structure for a research paper, and interpreting results. We have reviewed all AI-assisted work. All AI-generated suggestions on research methodology, experiments, paper structure, and readability were manually examined by the authors and used only as auxiliary feedback rather than as final decisions. AI-assisted code was reviewed, debugged, and tested by the authors to ensure correctness and consistency with our intended implementation. For literature identification and information searching, we verified the relevance and accuracy of the suggested references by checking the original papers and sources. All final experimental designs, result interpretations, and conclusions were independently determined and validated by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

8 REPRODUCIBILITY STATEMENT

We have made efforts to ensure the reproducibility of our results. The threat model, problem formulation, and overall attack pipeline are described in Section 3 and 4, while the key algorithmic details are presented in Appendix A. The experimental setup, including datasets, model architectures, unlearning methods, evaluation metrics, and implementation settings, is reported in Section 5.1. Additional details are provided in Appendix B. We will also include anonymized source code, configuration files, and scripts in the supplementary materials to facilitate independent reproduction of the main experimental results.

REFERENCES

Abeer Matar A Almalky, Ranyang Zhou, Shaahin Angizi, and Adnan Siraj Rakin. How vulnerable are large language models (llms) against adversarial bit-flip attacks? In *Proceedings of the Great Lakes Symposium on VLSI 2025*, pp. 534–539, 2025.

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.

Ruchika Chavhan, Da Li, and Timothy Hospedales. Conceptprune: Concept editing in diffusion models via skilled neuron pruning. *arXiv preprint arXiv:2405.19237*, 2024.

Huili Chen, Cheng Fu, Jishen Zhao, and Farinaz Koushanfar. Proflip: Targeted trojan attack with progressive bit flips. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7698–7707. IEEE, 2021.

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

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.

Zachary Coalson, Jeonghyun Woo, Chris S Lin, Joyce Qu, Yu Sun, Shiyang Chen, Lishan Yang, Gururaj Saileshwar, Prashant Nair, Bo Fang, et al. Prisonbreak: Jailbreaking large language models with at most twenty-five targeted bit-flips. *arXiv preprint arXiv:2412.07192*, 2024.

Zachary Coalson, AM Aahad, Stella Doehring, Zane Ma, and Sanghyun Hong. On the resilience of text-to-video diffusion models to hardware faults. *arXiv preprint arXiv:2608.29598*, 2026.

Sanjay Das, Swastik Bhattacharya, Souvik Kundu, Shamik Kundu, Anand Menon, Arnab Raha, and Kanad Basu. Attentionbreaker: Adaptive evolutionary optimization for unmasking vulnerabilities in llms through bit-flip attacks. *Transactions on Machine Learning Research*, 2025.

Aghyad Deeb and Fabien Roger. Do unlearning methods remove information from language model weights? *arXiv preprint arXiv:2410.08827*, 2024.

Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Chongyu Fan, Jiancheng Liu, Yihua Zhang, Eric Wong, Dennis Wei, and Sijia Liu. Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation. In *International Conference on Learning Representations*, volume 2024, pp. 53643–53673, 2024.

Chongyu Fan, Jinghan Jia, Yihua Zhang, Anil Ramakrishna, Mingyi Hong, and Sijia Liu. Towards llm unlearning resilient to relearning attacks: A sharpness-aware minimization perspective and beyond. *arXiv preprint arXiv:2502.05374*, 2025.

Chongyu Fan, Jiancheng Liu, Licong Lin, Jinghan Jia, Ruiqi Zhang, Song Mei, and Sijia Liu. Simplicity prevails: Rethinking negative preference optimization for llm unlearning. *Advances in Neural Information Processing Systems*, 38:1540–1567, 2026.

Jack Foster, Stefan Schoepf, and Alexandra Brintrup. Fast Machine Unlearning without Retraining through Selective Synaptic Dampening. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(11):12043–12051, March 2024. ISSN 2374-3468, 2159-5399. doi: 10.1609/aaai.v38i11.29092.

Rohit Gandikota, Joanna Materzynska, Jaden Fiotto-Kaufman, and David Bau. Erasing concepts from diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 2426–2436, 2023.

Rohit Gandikota, Hadas Orgad, Yonatan Belinkov, Joanna Materzyńska, and David Bau. Unified concept editing in diffusion models. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 5111–5120, 2024.

Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal Sunshine of the Spotless Net: Selective Forgetting in Deep Networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9301–9309, Seattle, WA, USA, June 2020. IEEE. ISBN 978-1-72817-168-5. doi: 10.1109/CVPR42600.2020.00932.

Chao Gong, Kai Chen, Zhipeng Wei, Jingjing Chen, and Yu-Gang Jiang. Reliable and efficient concept erasure of text-to-image diffusion models. In *European Conference on Computer Vision*, pp. 73–88. Springer, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified data removal from machine learning models. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

Jingkai Guo, Chaitali Chakrabarti, and Deliang Fan. Sbfa: Single sneaky bit flip attack to break large language models. *arXiv preprint arXiv:2509.21843*, 2025.

Jingkai Guo, Chaitali Chakrabarti, and Deliang Fan. Tfl: Targeted bit-flip attack on large language model. *arXiv preprint arXiv:2602.17837*, 2026.

SeungBum Ha, Saerom Park, and Sung Whan Yoon. Unlearning's blind spots: Over-unlearning and prototypical relearning attack. *arXiv preprint arXiv:2506.01318*, 2025.

Hasan Hassan, Yahya Can Tugrul, Jeremie S Kim, Victor Van der Veen, Kaveh Razavi, and Onur Mutlu. Uncovering in-dram rowhammer protection mechanisms: A new methodology, custom rowhammer patterns, and implications. In *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 1198–1213, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Shengyuan Hu, Yiwei Fu, Steven Wu, and Virginia Smith. Unlearning or obfuscating? jogging the memory of unlearned llms via benign relearning. In *International Conference on Learning Representations*, volume 2025, pp. 8857–8888, 2025.

Yichang Hu, Noah Brown, Yuhang Chen, Joshua Bakita, Tianlong Chen, Daniel Genkin, and Andrew Kwong. Gddrhammer: Greatly disturbing dram rows—cross-component rowhammer attacks from modern gpus. In *2026 IEEE Symposium on Security and Privacy (SP)*, pp. 546–564. IEEE, 2026.

Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. Knowledge unlearning for mitigating privacy risks in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14389–14408, 2023.

Patrick Jattke, Victor Van Der Veen, Pietro Frigo, Stijn Gunter, and Kaveh Razavi. Blacksmith: Scalable rowhammering in the frequency domain. In *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 716–734. IEEE, 2022.

Patrick Jattke, Max Wipfli, Flavien Solt, Michele Marazzi, Matej Bölcskei, and Kaveh Razavi. {ZenHammer}: Rowhammer attacks on {AMD} zen-based platforms. In *33rd USENIX Security Symposium (USENIX Security 24)*, pp. 1615–1633, 2024.

Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu. Flipping bits in memory without accessing them: An experimental study of dram disturbance errors. *ACM SIGARCH Computer Architecture News*, 42(3):361–372, 2014.

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

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

Andrew Kwong, Daniel Genkin, Daniel Gruss, and Yuval Yarom. Rambleed: Reading bits in memory without accessing them. In *2020 IEEE Symposium on Security and Privacy (SP)*, pp. 695–711. IEEE, 2020.

Nathaniel Li, Alexander Pan, Anjali Gopal, Summer Yue, Daniel Berrios, Alice Gatti, Justin D Li, Ann-Kathrin Dombrowski, Shashwat Goel, Long Phan, et al. The wmdp benchmark: Measuring and reducing malicious use with unlearning. *arXiv preprint arXiv:2403.03218*, 2024.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. Textbooks are all you need ii: phi-1.5 technical report. *arXiv preprint arXiv:2309.05463*, 2023.

Chris S Lin, Joyce Qu, and Gururaj Saileshwar. {GPUHammer}: Rowhammer attacks on {GPU} memories are practical. In *34th USENIX Security Symposium (USENIX Security 25)*, pp. 5719–5738, 2025.

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

Ping Liu and Chi Zhang. Erased or dormant? rethinking concept erasure through reversibility. *arXiv preprint arXiv:2505.16174*, 2025.

Aengus Lynch, Phillip Guo, Aidan Ewart, Stephen Casper, and Dylan Hadfield-Menell. Eight methods to evaluate robust unlearning in llms. *arXiv preprint arXiv:2402.16835*, 2024.

Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary C Lipton, and J Zico Kolter. Tofu: A task of fictitious unlearning for llms. *arXiv preprint arXiv:2401.06121*, 2024.

Muhammad Husnain Mubarik, Karthik Mohan Kumar, Pedro Antonio Pena, Keshavan Varadarajan, and Kunal Tyagi. From bit-position sensitivity to unequal error protection for dnn inference memory. *arXiv preprint arXiv:2607.19623*, 2026.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Adnan Siraj Rakin, Zhezhi He, and Deliang Fan. Bit-flip attack: Crushing neural network with progressive bit search. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1211–1220. IEEE, 2019.

Adnan Siraj Rakin, Zhezhi He, and Deliang Fan. Tbt: Targeted neural network attack with bit trojan. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13195–13204. IEEE, 2020.

Adnan Siraj Rakin, Zhezhi He, Jingtao Li, Fan Yao, Chaitali Chakrabarti, and Deliang Fan. T-bfa: Targeted bit-flip adversarial weight attack. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7928–7939, 2021.

Shoaib Ahmed Siddiqui, Adrian Weller, David Krueger, Gintare Karolina Dziugaite, Michael Mozer, and Eleni Triantafillou. From dormant to deleted: Tamper-resistant unlearning through weight-space regularization. *Advances in Neural Information Processing Systems*, 38:129326–129357, 2026.

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

Vinith Menon Suriyakumar, Rohan Alur, Ayush Sekhari, Manish Raghavan, and Ashia C Wilson. Unstable unlearning: The hidden risk of concept resurgence in diffusion models. In *ICLR 2025 Workshop on Navigating and Addressing Data Problems for Foundation Models*, 2024.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro Von Werra, Clémentine Fourrier, Nathan Habib, et al. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944*, 2023.

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.

Junpeng Wan, Yanan Guo, Zhi Zhang, Zhuo Li, Dave Jing Tian, and Zhenkai Zhang. Geforge: Hammering gddr memory to forge gpu page tables for fun and profit. In *2026 IEEE Symposium on Security and Privacy (SP)*, pp. 565–583. IEEE, 2026.

Jialai Wang, Ya Wen, Zhongmou Liu, Yuxiao Wu, Bingyi He, Zongpeng Li, and Ee-Chien Chang. Targeted bit-flip attacks on llm-based agents. *arXiv preprint arXiv:2603.10042*, 2026.

Jinqi Wen, Tong Xie, Runsheng Wang, and Meng Li. Drift: Harnessing inherent fault tolerance for efficient and reliable diffusion model inference. *arXiv preprint arXiv:2604.09073*, 2026.

Jing Wu, Trung Le, Munawar Hayat, and Mehrtash Harandi. Erasing undesirable influence in diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 28263–28273, 2025.

Haotian Xu, Qingsong Peng, Jie Shi, Huadi Zheng, Yu Li, and Cheng Zhuo. Silentstriker: Toward stealthy bit-flip attacks on large language models. *Advances in Neural Information Processing Systems*, 38:51981–52002, 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.

Jiyu Yang, Qiang He, Zheyu Zhou, Xiaohai Dai, Feifei Chen, Cong Tian, and Yun Yang. Edgethemis: Ensuring model integrity for edge intelligence. In *Proceedings of the ACM on Web Conference 2025*, pp. 3136–3146, 2025.

Fan Yao, Adnan Siraj Rakin, and Deliang Fan. {DeepHammer}: Depleting the intelligence of deep neural networks through targeted chain of bit flips. In *29th USENIX Security Symposium (USENIX Security 20)*, pp. 1463–1480, 2020.

Jin Yao, Eli Chien, Minxin Du, Xinyao Niu, Tianhao Wang, Zezhou Cheng, and Xiang Yue. Machine unlearning of pre-trained large language models. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 8403–8419, 2024.

Noureldin Zahran, Ahmad Tahmasivand, Ihsen Alouani, Khaled Khasawneh, and Mohammed Fouda. On jailbreaking quantized language models through fault injection attacks. In *Proceedings of the Great Lakes Symposium on VLSI 2025*, pp. 554–561, 2025.

Gong Zhang, Kai Wang, Xingqian Xu, Zhangyang Wang, and Humphrey Shi. Forget-me-not: Learning to forget in text-to-image diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1755–1764, 2024a.

Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. *arXiv preprint arXiv:2404.05868*, 2024b.

Appendix

A Pseudo Codes

Algorithm 1 BFRA: Bit-Flip Relearning Attack

Input: Unlearned model θ_{unl}, forget-related relearning set $\mathcal{D}_f^{\text{rel}}$, retain-related relearning set $\mathcal{D}_r^{\text{rel}}$, training loss function $\mathcal{L}$, maximum iterations T, candidate pool size K, budgets ϵ_r, ϵ_s, ϵ_c, stabilizer ρ
Output: Relearned model θ_{rel}
Initialize $\theta^0 \leftarrow \theta_{\text{unl}}$
Initialize $n^l_{\text{candidate}} \leftarrow 0$ and $n^l_{\text{final}} \leftarrow 0$ for each layer l
for $t = 0$ to $T - 1$ **do**
  $(\mathcal{P}, \mathcal{B}_f^t, \mathcal{B}_r^t) \leftarrow$ First-Order-Candidate-Selection $(\theta^t, \mathcal{D}_f^{\text{rel}}, \mathcal{D}_r^{\text{rel}}, \mathcal{L}, K, \epsilon_r, \rho)$
  if $\mathcal{P} = \emptyset$ **then**
    break
  end if
  $b^* \leftarrow$ Trial-Loss-Bit-Picking $(\theta^t, \theta^0, \mathcal{P}, \mathcal{B}_f^t, \mathcal{B}_r^t, \mathcal{L}, \epsilon_s, \epsilon_c)$
  if $b^* = \emptyset$ **then**
    break
  end if
  Permanently flip b^* to obtain θ^{t+1}
  Update $n^l_{\text{final}} \leftarrow n^l_{\text{final}} + 1$ for the layer containing b^*
end for
$\theta_{\text{rel}} \leftarrow \theta^t$
return θ_{rel}

B Experimental Settings

In this section, we detail all the hyperparameters of our experiments for easy reproducibility. For the same model-dataset setting, training and retraining-from-scratch processes share the same hyperparameters.

For image classifier tasks, training and retraining settings as detailed in Table B1. '$-$' indicates that the corresponding hyperparameter is not used.

Table B1: Training hyperparameters for original and retrain model in image classification tasks.

Dataset	Model	Batch size	Optimizer	lr	momentum	lr decay step	gamma	wd	epoch
CIFAR-10	AllCNN	64	SGD	0.1	0.9	-	-	-	30
	ResNet18	128	SGD	0.01	0.9	-	-	0.0005	200
	ViT	64	Adam	0.0001	-	-	-	0.0001	50
CIFAR-100	AllCNN	128	Adam	0.001	-	10	0.0001	-	300
	ResNet18	64	SGD	0.01	0.9	-	-	0.0005	200
	ViT	64	AdamW	0.0005	-	-	-	0.05	300

For image generation tasks in DDPM, we use a 1,000-step conditional diffusion model and fine-tune the pretrained model for 1,000 iterations with Adam (10^{-4}). For evaluation, we generate 500 images per class using DPM-Solver++ (second-order, 20 sampling steps, classifier-free guidance scale 2.0) and report classifier accuracy and FID on the forget and retain sets.

For LLM unlearning experiments on TOFU, we first fine-tune Llama-3.2-1B-Instruct, Llama-2-7B-chat models and Phi-1.5 on the full TOFU dataset to obtain the corresponding full models. These models are trained using supervised fine-tuning with the same optimization configuration. Specifically, we use the paged AdamW 32-bit optimizer with a learning rate of 1×10^{-5} (2×10^{-5}

Algorithm 2 First-Order Candidate Selection

Input: Current model θ^t, forget-related relearning set $\mathcal{D}_f^{\mathrm{rel}}$, retain-related relearning set $\mathcal{D}_r^{\mathrm{rel}}$, training loss function $\mathcal{L}$, candidate pool size K, budget ϵ_r, stabilizer ρ
Output: Candidate pool $\mathcal{P}$, mini-batches $\mathcal{B}_f^t$ and $\mathcal{B}_r^t$
Sample mini-batches $\mathcal{B}_f^t \subset \mathcal{D}_f^{\mathrm{rel}}$ and $\mathcal{B}_r^t \subset \mathcal{D}_r^{\mathrm{rel}}$
Compute $g^f \leftarrow \nabla_{\theta^t}\mathcal{L}(\theta^t; \mathcal{B}_f^t)$ and $g^r \leftarrow \nabla_{\theta^t}\mathcal{L}(\theta^t; \mathcal{B}_r^t)$
Initialize $\mathcal{P} \leftarrow \emptyset$
for each layer l **do**
  Compute $r_l \leftarrow \frac{n^l_{\mathrm{final}}+1}{n^l_{\mathrm{candidate}}+2}$ and $K_l \leftarrow K \cdot \frac{n_l r_l}{\sum_j n_j r_j}$
  For each weight w_i in layer l, compute

$$\Delta w_i^{\min} = -\frac{\epsilon_r}{|g_i^r| + \rho}, \quad \Delta w_i^{\max} = \frac{\epsilon_r}{|g_i^r| + \rho},$$

$$U_i^f = \max_{\Delta w \in [\Delta w_i^{\min}, \Delta w_i^{\max}]} (-g_i^f \Delta w)$$

  Sort weights in layer l by U_i^f in descending order
  for each sorted weight w_i in layer l **do**
    $\tau^f \leftarrow \min_{b_{j,k} \in \mathcal{P}}(-g_j^f \Delta w_{j,k})$ if $|\mathcal{P}| = K$
    if $|\mathcal{P}| = K$ and $U_i^f \le \tau^f$ **then**
      break
    end if
    for each sign or exponent bit $b_{i,k}$ of w_i **do**
      Compute $\Delta w_{i,k}$
      Skip $b_{i,k}$ if $|g_i^r \Delta w_{i,k}| > \epsilon_r$ or the quota K_l is exceeded
      Let b_τ denote the candidate in $\mathcal{P}$ with value τ^f
      Update $\mathcal{P}$ by

$$\mathcal{P} \leftarrow \begin{cases} \mathcal{P} \cup \{b_{i,k}\}, & |\mathcal{P}| < K, \\ \mathcal{P} \setminus \{b_\tau\} \cup \{b_{i,k}\}, & |\mathcal{P}| = K \text{ and } -g_i^f \Delta w_{i,k} > \tau^f, \\ \mathcal{P}, & \text{otherwise.} \end{cases}$$

      if $b_{i,k}$ is newly inserted into $\mathcal{P}$ **then**
        $n^l_{\mathrm{candidate}} \leftarrow n^l_{\mathrm{candidate}} + 1$
      end if
      Update $\tau^f \leftarrow \min_{b_{j,k} \in \mathcal{P}}(-g_j^f \Delta w_{j,k})$
    end for
  end for
end for
return $\mathcal{P}, \mathcal{B}_f^t, \mathcal{B}_r^t$

for Phi-1.5), a weight decay of 0.01, and a linear warmup and decay learning rate schedule. The models are trained for 5 epochs with 1 warmup epoch. Training is conducted on 2 GPUs with a per-device batch size of 4 and gradient accumulation steps of 4, resulting in an effective batch size of 32. We use BF16 precision, FlashAttention-2, and gradient checkpointing to improve training efficiency. All experiments are performed with DeepSpeed ZeRO Stage-3 distributed training and a maximum sequence length of 512 tokens. The random seed is set to 0 for reproducibility.

For LLM unlearning experiments on WMDP, we use the Zephyr-7B-BETA model as the target model, following the standard evaluation protocol of the WMDP benchmark.

Across all three tasks, BFRA performs at most 100 bit-flipping rounds, accepting one bit flip per round. We set the candidate pool size to Top-$K = 128$ and restrict bit flips to the sign and exponent bits. The task-specific configurations differ mainly in the batch settings for candidate-gradient computation and trial-loss evaluation, as well as the retain-utility constraints. For LLM experiments,

Algorithm 3 Trial-Loss Bit Picking

Input: Current model θ^t, initial model θ^0, candidate pool $\mathcal{P}$, mini-batches $\mathcal{B}_f^t$ and $\mathcal{B}_r^t$, training loss function $\mathcal{L}$, budgets ϵ_s and ϵ_c
Output: Selected bit b^*, or $\emptyset$ if no candidate satisfies the retain-loss constraints
for each candidate bit $b_{i,k} \in \mathcal{P}$ **do**
  Compute the perturbation $\Delta w_{i,k}$ induced by flipping $b_{i,k}$
  Compute

$$\Delta\mathcal{L}_{i,k}^f = \mathcal{L}(\theta^t + \Delta w_{i,k}\mathbf{e}_i; \mathcal{B}_f^t) - \mathcal{L}(\theta^t; \mathcal{B}_f^t)$$

end for
Sort $\mathcal{P}$ in ascending order of $\Delta\mathcal{L}_{i,k}^f$
Initialize $b^* \leftarrow \emptyset$
for each sorted candidate bit $b_{i,k} \in \mathcal{P}$ **do**
  Compute the perturbation $\Delta w_{i,k}$ induced by flipping $b_{i,k}$
  Compute

$$\Delta\mathcal{L}_{i,k}^{r,s} = \mathcal{L}(\theta^t + \Delta w_{i,k}\mathbf{e}_i; \mathcal{B}_r^t) - \mathcal{L}(\theta^t; \mathcal{B}_r^t)$$

  Compute

$$\Delta\mathcal{L}_{i,k}^{r,c} = \mathcal{L}(\theta^t + \Delta w_{i,k}\mathbf{e}_i; \mathcal{B}_r^t) - \mathcal{L}(\theta^0; \mathcal{B}_r^t)$$

  if $\Delta\mathcal{L}_{i,k}^{r,s} \leq \epsilon_s$ and $\Delta\mathcal{L}_{i,k}^{r,c} \leq \epsilon_c$ **then**
    $b^* \leftarrow b_{i,k}$
    break
  end if
end for
return b^*

we use four $\mathcal{D}_f$ batches with a batch size of 1 to compute candidate gradients in each round, and 16 $\mathcal{D}_f$ batches for trial-loss evaluation and both retain constraints. We set $\epsilon_r = 0.8$, $\epsilon_c = 0.8$, and $\epsilon_s = 0.01$, with the random batch seed fixed to 42. For DDPM experiments on CIFAR-10, one $\mathcal{D}_f$ batch with a batch size of 64 is used for candidate-gradient computation, while one batch is used for trial-loss evaluation and both retain constraints; the corresponding thresholds are $\epsilon_r = 2.4$, $\epsilon_c = 3.2$, and $\epsilon_s = 0.04$. For image-classification experiments, we use four $\mathcal{D}_f$ batches with a batch size of 64 for candidate-gradient computation and 16 $\mathcal{D}_f$ batches for trial-loss evaluation and both retain constraints, with $\epsilon_r = 0.4$, $\epsilon_c = 0.8$, and $\epsilon_s = 0.01$.

C Extended Experiments

In this section, we provide additional experiments to further evaluate the effectiveness, robustness, and generality of BFRA. We first extend the unlearning results to more image-classification settings, including CIFAR-100, different target classes, sequential unlearning, and random unlearning, and further report extended results on text generation. We then present detailed relearning trajectories and a series of ablation studies to analyze the effects of different stages, individual components, layer quota, bit-field selection, and simultaneous bit flipping. In addition, we study the sensitivity to key hyperparameters, evaluate membership inference attacks on image-classification models, and visualize the distributions of selected bits and their contributions. Finally, we compare relearning behavior under certified removal and different machine-unlearning strengths to further examine the applicability of BFRA under diverse unlearning settings.

C.1 Extended Unlearning Results for Image Classification

We further extend the class-unlearning experiments to CIFAR-100 across different model architectures and unlearning methods. As shown in Table C2, BFRA consistently recovers the forgotten class across AllCNN, ResNet-18, and ViT, with FA increasing to nearly 100% in most settings after only a small number of bit flips. Meanwhile, the retain and test performance remain largely preserved, although the exact utility degradation and required flipping rounds T vary across architectures and

unlearning methods. Overall, these results demonstrate that the bit-level relearning vulnerability is not specific to a particular model or unlearning algorithm, but generalizes broadly to different image classifiers and class-unlearning settings.

Extended Class Unlearning Results on CIFAR-100

Table C2: Class unlearning results on image classification models on CIFAR-100. † indicates BFRA-relearned results.

Model	**AllCNN, CIFAR-100**				**ResNet-18, CIFAR-100**				**ViT, CIFAR-100**			
Metric	FA	RA	TA	T	FA	RA	TA	T	FA	RA	TA	T
Original	$100.0_{\pm 0.00}$	$99.98_{\pm 0.00}$	$61.41_{\pm 0.00}$	-	$100.0_{\pm 0.00}$	$99.98_{\pm 0.00}$	$76.12_{\pm 0.00}$	-	$99.40_{\pm 0.00}$	$96.97_{\pm 0.00}$	$65.04_{\pm 0.00}$	-
Retrain	$0.00_{\pm 0.00}$	$99.98_{\pm 0.00}$	$62.53_{\pm 0.00}$	-	$0.00_{\pm 0.00}$	$99.98_{\pm 0.00}$	$78.31_{\pm 0.00}$	-	$0.00_{\pm 0.00}$	$96.96_{\pm 0.00}$	$65.10_{\pm 0.00}$	-
GA	$0.06_{\pm 0.02}$	$92.80_{\pm 0.06}$	$51.89_{\pm 0.09}$	-	$0.60_{\pm 0.20}$	$97.64_{\pm 0.43}$	$69.35_{\pm 0.50}$	-	$0.80_{\pm 0.20}$	$88.24_{\pm 0.36}$	$57.64_{\pm 0.62}$	-
GA†	$100.0_{\pm 0.00}$	$93.62_{\pm 0.09}$	$51.49_{\pm 0.10}$	15	$100.0_{\pm 0.00}$	$97.05_{\pm 0.13}$	$68.91_{\pm 0.12}$	10	$100.0_{\pm 0.00}$	$85.34_{\pm 0.45}$	$55.59_{\pm 0.32}$	58
BS	$1.20_{\pm 0.28}$	$85.88_{\pm 0.21}$	$47.26_{\pm 0.45}$	-	$0.20_{\pm 0.06}$	$90.59_{\pm 0.36}$	$60.51_{\pm 1.08}$	-	$3.00_{\pm 1.02}$	$74.23_{\pm 1.31}$	$46.06_{\pm 1.10}$	-
BS†	$100.0_{\pm 0.00}$	$91.92_{\pm 0.24}$	$50.84_{\pm 0.33}$	15	$100.0_{\pm 0.00}$	$95.77_{\pm 1.08}$	$64.99_{\pm 0.94}$	15	$100.0_{\pm 0.00}$	$81.52_{\pm 0.26}$	$51.32_{\pm 0.35}$	10
SalUn	$0.40_{\pm 0.26}$	$75.16_{\pm 1.82}$	$44.68_{\pm 2.14}$	-	$0.60_{\pm 0.14}$	$97.64_{\pm 0.25}$	$68.86_{\pm 0.18}$	-	$3.70_{\pm 0.81}$	$84.72_{\pm 0.96}$	$54.79_{\pm 1.38}$	-
SalUn†	$99.80_{\pm 0.04}$	$82.23_{\pm 0.05}$	$47.54_{\pm 0.08}$	20	$100.0_{\pm 0.00}$	$94.06_{\pm 0.20}$	$64.17_{\pm 0.31}$	25	$100.0_{\pm 0.00}$	$82.54_{\pm 0.28}$	$53.80_{\pm 0.31}$	20
FT	$0.00_{\pm 0.00}$	$94.14_{\pm 0.17}$	$53.28_{\pm 0.32}$	-	$1.20_{\pm 0.42}$	$97.90_{\pm 0.24}$	$64.59_{\pm 0.57}$	-	$0.00_{\pm 0.00}$	$87.87_{\pm 0.52}$	$58.35_{\pm 0.60}$	-
FT†	$100.0_{\pm 0.00}$	$81.80_{\pm 0.35}$	$48.87_{\pm 0.63}$	35	$97.00_{\pm 0.34}$	$94.78_{\pm 0.52}$	$61.06_{\pm 0.66}$	15	$92.80_{\pm 0.10}$	$79.95_{\pm 1.32}$	$53.19_{\pm 0.71}$	60
SCRUB	$0.00_{\pm 0.00}$	$99.98_{\pm 0.00}$	$61.46_{\pm 0.02}$	-	$0.00_{\pm 0.00}$	$99.93_{\pm 0.02}$	$75.79_{\pm 0.07}$	-	$0.00_{\pm 0.00}$	$90.42_{\pm 0.21}$	$58.42_{\pm 0.37}$	-
SCRUB†	$100.0_{\pm 0.00}$	$99.49_{\pm 0.02}$	$59.08_{\pm 0.04}$	5	$100.0_{\pm 0.00}$	$93.14_{\pm 0.54}$	$64.21_{\pm 0.39}$	65	$100.0_{\pm 0.00}$	$84.53_{\pm 0.46}$	$53.45_{\pm 0.51}$	30

Extended Class Unlearning Results on Different Target Classes

To evaluate whether BFRA generalizes beyond a specific unlearning target, we further conduct class-unlearning experiments on multiple target classes, as shown in Table C3. Across different target classes, datasets, and architectures, BFRA consistently restores the forgotten class to a high level, while largely preserving or even improving retain and test accuracy. The same overall pattern is observed for Classes 3, 5, 7, and 9, indicating that the relearning capability of BFRA is not tied to a particular target class. Moreover, strong recovery can typically be achieved with only a small number of bit flips, especially on CIFAR-100 models. These results demonstrate that BFRA provides robust and class-agnostic relearning across diverse class-unlearning settings.

Table C3: Class unlearning with different target classes on image classification models. † indicates BFRA-relearned results.

Model		**AllCNN, CIFAR-10**				**AllCNN, CIFAR-100**				**ResNet18, CIFAR-100**			
Class	Metric	FA	RA	TA	T	FA	RA	TA	T	FA	RA	TA	T
3	GA	0.20	89.42	78.21	-	0.00	90.74	50.11	-	0.80	95.74	68.14	-
	GA†	99.88	90.10	76.08	60	100.0	97.07	54.58	15	100.0	96.09	68.28	25
5	GA	0.66	75.74	66.26	-	0.00	88.66	49.05	-	1.80	86.99	61.71	-
	GA†	94.32	86.90	74.93	26	100.0	94.91	52.66	15	100.0	94.41	66.46	15
7	GA	0.00	93.26	78.06	-	0.00	90.80	50.58	-	2.80	91.24	64.53	-
	GA†	99.90	92.10	76.96	37	100.0	95.78	54.30	15	100.0	95.89	68.00	20
9	GA	0.06	88.55	73.83	-	0.60	80.46	44.76	-	4.40	83.03	58.48	-
	GA†	100.00	92.45	76.41	50	100.0	92.09	51.04	15	100.0	90.40	63.51	20

Extended Results on Sequential Unlearning

To further evaluate BFRA in a practical setting, we extend our experiments to sequential class unlearning, where forgetting requests arrive successively. As shown in Table C4, BFRA consistently restores the forgotten knowledge after each unlearning stage. As more classes are sequentially removed, the task becomes harder and generally requires more bit flips, but BFRA still achieves strong

recovery while maintaining comparable retain and test utility. These results show that BFRA remains effective under accumulated sequential forgetting requests.

Table C4: Sequential class unlearning on AllCNN, CIFAR10, GA. † indicates BFRA-relearned results.

Model	Unlearned				Relearned			
Class	FA	RA	TA	T	FA	RA	TA	T
1	0.00	88.21	73.47	-	100.0	93.76	77.28	28
1+7	2.54	91.92	77.23	-	99.92	89.86	74.20	100
1+7+3	0.87	78.87	70.04	-	98.33	88.11	72.70	70
1+7+3+9	0.11	79.98	71.82	-	97.51	86.34	71.45	100

Extended Random Unlearning Results

To further evaluate the generality of BFRA beyond class unlearning, we extend our experiments to random unlearning, where 10% of the CIFAR-10 training samples are randomly selected for forgetting. We report results averaged on 5 random runs, which makes the target forgetting dataset different in each run. As shown in Table C5, BFRA consistently recovers the forgotten samples across different unlearning methods and model architectures, showing that its relearning capability is not specific to class-wise forgetting, although the recovery amount and the resulting utility trade-off vary across unlearning methods. In most cases, substantial relearning is achieved with only tens of bit flips while maintaining utility. These results demonstrate that BFRA also generalizes well to random unlearning.

Table C5: Random unlearning results on CIFAR-10 (forget ratio=10%) across AllCNN, ResNet-18, and ViT. † indicates BFRA-relearned variants.

Method	**AllCNN, CIFAR-10**				**ResNet-18, CIFAR-10**				**ViT, CIFAR-10**			
Metric	FA	RA	TA	T	FA	RA	TA	T	FA	RA	TA	T
Original	$99.72_{\pm 0.53}$	$99.68_{\pm 0.10}$	$84.20_{\pm 0.28}$	-	$99.86_{\pm 0.00}$	$99.84_{\pm 0.00}$	$92.50_{\pm 0.00}$	-	$95.35_{\pm 0.00}$	$94.32_{\pm 0.00}$	$79.76_{\pm 0.00}$	-
Retrain	$83.94_{\pm 0.16}$	$99.71_{\pm 0.09}$	$83.24_{\pm 0.31}$	-	$83.54_{\pm 0.00}$	$99.92_{\pm 0.00}$	$93.17_{\pm 0.00}$	-	$89.74_{\pm 0.00}$	$95.65_{\pm 0.00}$	$79.70_{\pm 0.00}$	-
GA	$99.32_{\pm 0.15}$	$99.32_{\pm 0.13}$	$82.65_{\pm 0.41}$	-	$93.22_{\pm 0.29}$	$93.99_{\pm 0.43}$	$86.76_{\pm 0.53}$	-	$92.74_{\pm 0.43}$	$93.94_{\pm 0.22}$	$76.79_{\pm 0.36}$	-
GA†	$99.01_{\pm 0.13}$	$99.45_{\pm 0.08}$	$82.79_{\pm 0.16}$	25	$95.34_{\pm 0.33}$	$95.28_{\pm 0.34}$	$88.41_{\pm 0.45}$	52	$94.01_{\pm 0.02}$	$94.25_{\pm 0.03}$	$77.13_{\pm 0.04}$	90
BS	$98.48_{\pm 0.36}$	$99.10_{\pm 0.18}$	$82.42_{\pm 0.44}$	-	$98.90_{\pm 0.46}$	$99.05_{\pm 0.19}$	$91.70_{\pm 0.23}$	-	$93.38_{\pm 0.45}$	$94.09_{\pm 0.19}$	$78.92_{\pm 0.34}$	-
BS†	$99.21_{\pm 0.14}$	$98.98_{\pm 0.23}$	$82.31_{\pm 0.46}$	18	$98.92_{\pm 0.02}$	$93.85_{\pm 0.54}$	$85.58_{\pm 0.62}$	44	$94.98_{\pm 0.18}$	$94.30_{\pm 0.12}$	$77.00_{\pm 0.05}$	6
SalUn	$98.34_{\pm 0.35}$	$98.17_{\pm 0.31}$	$81.45_{\pm 0.54}$	-	$93.56_{\pm 0.31}$	$93.18_{\pm 0.49}$	$85.16_{\pm 0.66}$	-	$93.39_{\pm 0.34}$	$94.54_{\pm 0.30}$	$78.99_{\pm 0.42}$	-
SalUn†	$99.24_{\pm 0.16}$	$99.16_{\pm 0.17}$	$82.44_{\pm 0.42}$	29	$95.88_{\pm 0.45}$	$95.15_{\pm 0.17}$	$87.41_{\pm 0.28}$	42	$94.39_{\pm 0.03}$	$95.42_{\pm 0.16}$	$78.02_{\pm 0.31}$	9
SCRUB	$85.45_{\pm 0.18}$	$99.38_{\pm 0.01}$	$83.02_{\pm 0.03}$	-	$90.76_{\pm 0.19}$	$97.14_{\pm 0.30}$	$87.38_{\pm 0.50}$	-	$90.86_{\pm 0.14}$	$95.28_{\pm 0.15}$	$79.01_{\pm 0.28}$	-
SCRUB†	$97.98_{\pm 0.42}$	$97.12_{\pm 0.33}$	$80.85_{\pm 0.48}$	60	$96.96_{\pm 0.10}$	$97.02_{\pm 0.15}$	$86.89_{\pm 0.43}$	32	$94.45_{\pm 0.03}$	$94.89_{\pm 0.21}$	$78.41_{\pm 0.24}$	26

C.2 Extended Unlearning Results for Text Generation

To further evaluate BFRA on LLM unlearning, we extend the experiments to different datasets and forgetting settings. TOFU and WMDP differ in data structure: TOFU considers QA-pair forgetting, while WMDP evaluates domain-level forgetting over bio and cyber knowledge. As shown in Tables C6 and C7, BFRA consistently recovers forgotten knowledge across different LLM architectures and unlearning methods on TOFU. A similar pattern is observed on WMDP: BFRA-bio and BFRA-cyber, which use only bio or cyber data for relearning, respectively, substantially restore the corresponding forgotten knowledge with only a small number of bit flips. These results show that BFRA is effective across different datasets, knowledge formats, and forgetting settings.

Table C6: LLM relearning results on TOFU-Forget5%. † indicates BFRA-relearned results.

Model	**Llama3.2-1B**					**Llama2-7B**					**Phi-1.5B**				
Metric	Prob	ROUGE	MU	FQ	T	Prob	ROUGE	MU	FQ	T	Prob	ROUGE	MU	FQ	T
original	0.8854	0.8265	0.5989	6.5e-13	-	0.9894	0.9659	0.6341	5.9e-14	-	0.9238	0.7961	0.3910	3.1e-12	-
retrain	0.1268	0.3851	0.5990	1.0000	-	0.1491	0.3993	0.6334	1.0000	-	0.1213	0.3845	0.4162	1.0000	-
GA	0.1841	0.3162	0.4428	0.1841	-	0.2205	0.3936	0.5394	4.6e-7	-	0.2190	0.4693	0.3637	1.1e-5	-
GA†	0.3833	0.3919	0.4700	4.6e-7	20	0.5857	0.4669	0.5986	4.5e-8	100	0.4867	0.4488	0.3587	1.2e-6	60
NPO	0.2568	0.3084	0.5255	0.1123	-	0.1238	0.3011	0.4504	1.1e-5	-	0.0541	0.3583	0.3784	0.7126	-
NPO†	0.4382	0.4194	0.5233	8.1e-8	30	0.5407	0.4119	0.5801	8.1e-7	40	0.2718	0.3913	0.2975	0.1123	60
RMU	0.0166	0.1100	0.5622	6.7e-7	-	0.0253	0.1519	0.5949	0.0006	-	0.0740	0.2895	0.3204	0.9238	-
RMU†	0.4412	0.4185	0.5132	2.4e-8	40	0.7490	0.5182	0.6207	1.4e-6	50	0.6816	0.5352	0.4085	1.4e-6	50
simNPO	0.2468	0.3490	0.4048	7.5e-5	-	0.0271	0.2549	0.5804	0.9238	-	0.3378	0.4520	0.4198	3e-5	-
simNPO†	0.3237	0.3915	0.3999	2.6e-8	20	0.2568	0.2861	0.5044	0.0297	90	0.4383	0.4678	0.4064	0.0003	60

Table C7: LLM relearning results on WMDP. † indicates BFRA-relearned results.

Metrics	$\mathrm{FA}_{\mathrm{bio}}$	$\mathrm{FA}_{\mathrm{cyber}}$	$\mathrm{RA}_{\mathrm{MMLU}}$	T
Original	0.6442	0.4432	0.5853	-
RMU unlearned	0.3111	0.2723	0.5748	-
RMU + BFRA-bio	0.5719	0.3578	0.5491	20
RMU + BFRA-cyber	0.4894	0.3946	0.5510	10

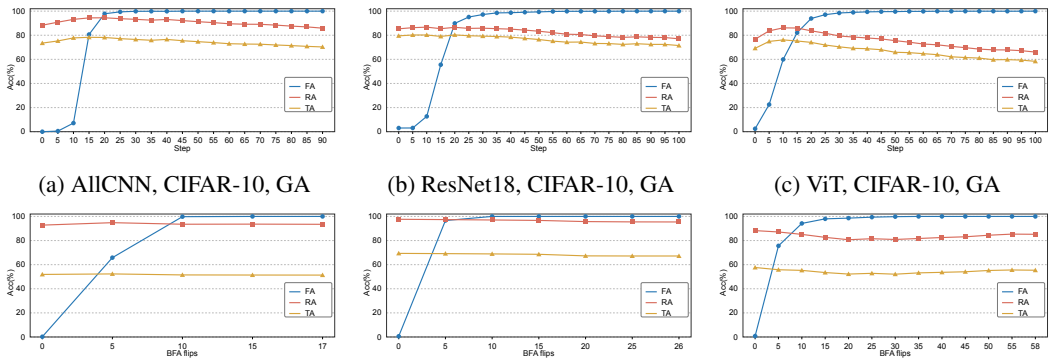
We further evaluate BFRA on parameter-efficient unlearning by applying MU baselines using Low-Rank Adaptation (LoRA) to Llama3.2-1B and performing BFRA on the resulting LoRA-unlearned models. As shown in Table C8, BFRA consistently recovers forgotten knowledge across different unlearning methods, with clear improvements in Prob and ROUGE while maintaining comparable retain utility. These results show that the bit-level relearning vulnerability also persists under LoRA-based parameter-efficient unlearning.

Table C8: LLM relearning results on TOFU-Forget1% using LoRA PEFT. † indicates BFRA-relearned results.

Model	**Llama3.2-1B, TOFU-Forget1%, LoRA**				
Metric	Prob	ROUGE	MU	FQ	T
original	0.9011	0.8731	0.6033	0.0100	-
retrain	0.1655	0.4071	0.6054	1.0000	-
GA	0.1983	0.3715	0.5663	0.1650	-
GA†	0.7655	0.6621	0.5885	0.0030	10
NPO	0.1078	0.2970	0.5109	0.9188	-
NPO†	0.2705	0.3664	0.4537	0.1650	50
RMU	0.1862	0.3565	0.4586	0.0971	-
RMU†	0.8046	0.6395	0.5710	0.0068	20
simNPO	0.1575	0.2677	0.5332	0.9188	-
simNPO†	0.2631	0.3770	0.4768	0.2657	40

C.3 Extended Relearning Trajectories

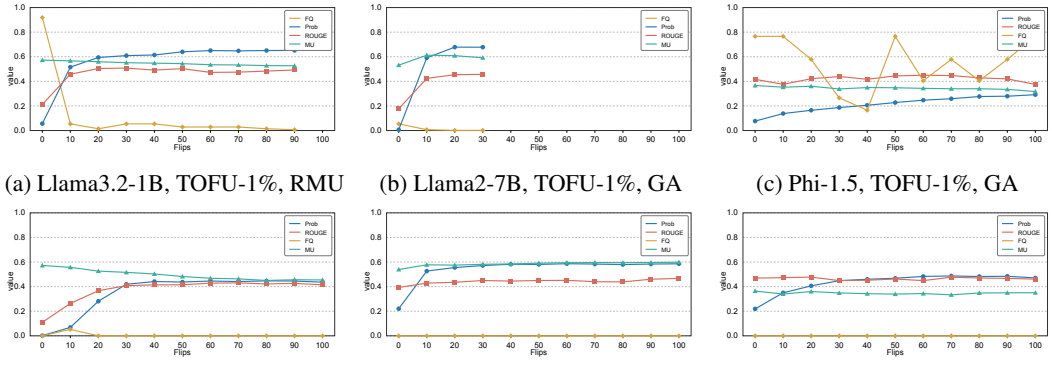
We further provide extended BFRA trajectories in the appendix to examine its behavior across different settings. Despite variations in models and unlearning configurations, the trajectories exhibit a consistent pattern: a small number of early bit flips account for most of the relearning gain, while subsequent flips mainly refine the recovery and may gradually affect retain utility. Importantly, the best recovery–utility trade-off is often reached well before exhausting the full bit-flip budget, indicating that effective relearning is concentrated in a small set of high-impact parameter bits. The consistent trajectory patterns across settings further support the stability and bit efficiency of BFRA.



(a) AllCNN, CIFAR-10, GA (b) ResNet18, CIFAR-10, GA (c) ViT, CIFAR-10, GA

(d) AllCNN, CIFAR-100, GA (e) ResNet18, CIFAR-100, GA (f) ViT, CIFAR-100, GA

Figure C1: Relearning Trajectories on Image Classification.



(a) Llama3.2-1B, TOFU-1%, RMU (b) Llama2-7B, TOFU-1%, GA (c) Phi-1.5, TOFU-1%, GA

(d) Llama3.2-1B, TOFU-5%, RMU (e) Llama2-7B, TOFU-5%, GA (f) Phi-1.5, TOFU-5%, GA

Figure C2: Relearning Trajectories on LLMs.

C.4 Extended Ablations

Stage-wise Ablation

BFRA consists of two stages: first-order candidate selection followed by trial-loss bit picking (TL). We ablate the second stage to evaluate its necessity. Ablating the first stage is impractical, since directly evaluating the loss over all model bits would incur prohibitive computational cost. As shown in Tables C9 and C10, removing TL consistently leads to highly unstable behavior across both image classifiers and LLMs. Although some settings can still recover the forgetting metric, retain and test utility often collapse severely; in other cases, relearning itself fails. Moreover, BFRA without TL frequently stops after only a small number of flips despite poor recovery, indicating abnormal premature stopping rather than successful optimization.

This behavior arises because a bit flip can induce a relatively large parameter perturbation, violating the local-perturbation assumption underlying the first-order Taylor approximation. Consequently, the loss change predicted by the gradient can differ substantially from the true loss change, and may even have the opposite direction. Directly accepting bits that appear effective under the first-order estimate can therefore move the model toward undesirable states and quickly exhaust the available search directions. The TL stage explicitly evaluates the actual loss after each candidate flip, filtering out such misleading candidates and substantially improving the stability and effectiveness of BFRA.

Table C9: Stage-wise ablation on image classification models. TL is short for Trial Loss Bit Picking stage.

Model	**AllCNN, CIFAR-10**				**ResNet-18, CIFAR-10**				**ViT, CIFAR-10**			
Metric	FA	RA	TA	T	FA	RA	TA	T	FA	RA	TA	T
Original	99.48	99.73	84.20	-	99.96	99.84	92.50	-	94.66	92.52	79.37	-
GA+BFRA	100.0	93.76	77.28	28	99.10	85.03	78.34	39	99.96	73.45	65.06	58
GA+BFRA(no TL)	98.40	76.29	62.97	35	100.00	2.84	2.49	40	100.00	20.82	18.59	35
BS+BFRA	99.98	92.45	74.48	43	100.0	88.02	80.28	40	99.56	75.94	66.84	65
BS+BFRA(no TL)	98.12	63.34	54.21	30	100.0	1.51	1.43	50	100.0	4.96	4.87	30
SalUn+BFRA	99.98	93.24	69.13	27	100.0	97.02	88.19	8	100.0	70.44	61.84	74
SalUn+BFRA(no TL)	0.00	13.54	13.14	5	100.0	61.30	55.39	30	100.0	10.18	9.52	30
FT+BFRA	99.30	88.80	74.38	49	98.02	87.02	78.19	53	81.76	87.16	75.53	13
FT+BFRA(no TL)	99.30	88.80	74.38	40	100.0	18.02	16.27	100	99.90	35.58	32.70	30
SCRUB+BFRA	90.22	94.28	78.16	22	94.58	89.35	79.27	47	99.02	82.11	70.88	49
SCRUB+BFRA(no TL)	0.00	72.17	63.39	15	76.66	22.23	19.87	100	99.96	17.45	15.46	40

Table C10: Stage-wise ablation on LLMs with TOFU Forget-1%. TL is short for Trial Loss Bit Picking stage.

Model	**Llama3.2-1B**					**Phi-1.5**				
Metric	Prob	ROUGE	MU	FQ	T	Prob	ROUGE	MU	FQ	T
original	0.9011	0.8731	0.6033	0.0100	-	0.9233	0.8424	0.3910	0.0005	-
GA+BFRA	0.4024	0.4205	0.5019	0.0030	50	0.3572	0.4607	0.3421	0.2786	100
GA+BFRA(no TL)	0.0181	0.1036	0.4046	0.6385	20	0.0718	0.2763	0.1432	0.9188	10
NPO+BFRA	0.2494	0.3730	0.5284	0.4046	40	0.3829	0.4728	0.3425	0.0971	100
NPO+BFRA(no TL)	0.0001	0.0127	0.1430	0.0000	10	0.0707	0.1426	0.0168	0.7659	20
RMU+BFRA	0.6515	0.5079	0.5349	0.0068	50	0.6759	0.5214	0.3999	0.0541	100
RMU+BFRA(no TL)	0.0135	0.0612	0.5414	0.0000	10	0.0726	0.1141	0.5786	0.2731	10
simNPO+BFRA	0.9055	0.9046	0.5337	0.0143	20	0.5896	0.5123	0.3886	0.0013	100
simNPO+BFRA(no TL)	0.0175	0.0808	0.0971	0.0225	20	0.0000	0.0087	0.0000	0.0000	10

Fine-grained Component-wise Ablation

We further conduct fine-grained component-wise ablation studies on various model architectures and datasets, over the first-order constraint (FOC), trial-loss constraint (TLC), and layer quota (LQ). As shown in Table C11 and C12, although FOC and TLC are both designed to prevent excessive degradation of retain utility, FOC also plays an important role in enabling effective relearning. By bounding the magnitude of parameter changes, FOC better preserves the local-perturbation assumption required by the first-order Taylor approximation and stabilizes activation values, making the first-stage candidate screening more reliable. Consequently, configurations without FOC fail to achieve a desirable recovery–utility trade-off: they either provide little recovery or severely damage model utility, and often terminate abnormally after only a few flips due to unstable numerical behavior.

TLC and LQ alone provide limited relearning capability, but when combined with FOC, both substantially improve the recovery–utility trade-off by further suppressing harmful bit flips and constraining the search. Their joint use provides the most stable behavior across image classifiers, LLMs, and diffusion models. Overall, these results show that FOC provides the foundation for effective and stable bit selection, while TLC and LQ offer complementary protection against excessive utility degradation.

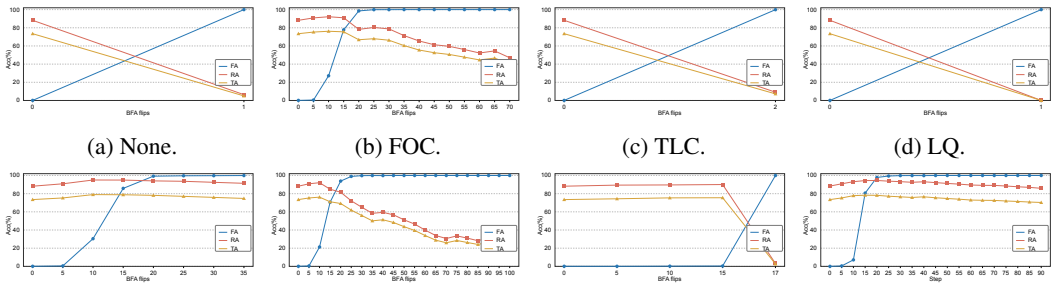
Table C11: Component-wise ablation on image classification. Here, FOC is short for First-Order Constraint on $\mathcal{D}_r^{\text{rel}}$; TLC is short for Trial-Loss Constraint on $\mathcal{D}_r^{\text{rel}}$; LQ is short for Layer Quota.

Module			**AllCNN, CIFAR-100**				**ResNet18, CIFAR-100**			
FOC	TLC	LQ	FA	RA	TA	T	FA	RA	TA	T
Unlearned			0.06	92.80	51.89	-	0.60	97.64	69.35	-
-	-	-	0.06	92.80	51.89	0	1.20	98.05	70.08	1
✓	-	-	100.0	62.51	35.07	30	100.0	41.72	28.03	35
-	✓	-	0.06	92.80	51.89	0	1.20	98.05	70.08	1
-	-	✓	100.0	0.00	0.00	2	100.0	0.00	0.00	1
✓	✓	-	100.0	93.63	51.65	15	100.0	97.34	69.33	10
✓	-	✓	100.0	73.57	41.58	10	100.0	41.05	27.55	30
-	✓	✓	0.06	92.80	51.89	0	0.60	97.64	69.35	0
✓	✓	✓	100.0	93.62	51.49	15	100.0	97.05	68.91	10

Table C12: Component-wise ablation on text generation task. Here, FOC is short for First-Order Constraint on $\mathcal{D}_r^{\text{rel}}$; TLC is short for Trial-Loss Constraint on $\mathcal{D}_r^{\text{rel}}$; LQ is short for Layer Quota.

Module			**Phi-1.5B, TOFU-1%**					**DDPM, CIFAR-10**				
FOC	TLC	LQ	Prob	ROUGE	MU	FQ	T	FA	FID_f	RA	FID_r	T
Unlearned			0.0768	0.4165	0.3660	0.7659	-	0.00	208.60	97.73	20.20	-
-	-	-	0.0768	0.4165	0.3660	0.7659	0	0.00	208.60	97.73	20.20	0
✓	-	-	0.2808	0.4317	0.2618	0.5786	70	8.60	117.71	97.22	21.01	2
-	✓	-	0.0768	0.4165	0.3660	0.7659	0	0.00	208.60	97.73	20.20	0
-	-	✓	0.0768	0.4165	0.3660	0.7659	0	0.00	494.83	11.04	556.42	18
✓	✓	-	0.1702	0.3765	0.3379	0.7659	20	14.60	247.92	84.31	49.80	4
✓	-	✓	0.2731	0.2895	0.0575	0.5786	80	87.88	68.55	87.04	25.99	80
-	✓	✓	0.0807	0.4183	0.3699	0.5786	4	0.02	212.55	94.30	23.06	50
✓	✓	✓	0.3572	0.4607	0.3421	0.2786	100	95.20	80.50	95.44	24.04	30

We further plot the relearning trajectories of the ablation results in Figure C3, revealing clear differences in relearning behavior. Without FOC, the search often terminates abnormally after only one or two bit flips due to numerical instability, so the resulting small T reflects premature failure rather than fast relearning. With FOC enabled, LQ generally accelerates convergence toward a favorable recovery–utility trade-off by distributing bit flips more effectively across layers. In contrast, adding TLC to FOC imposes a more conservative acceptance criterion. It effectively suppresses the degradation of retain utility, but may slightly reduce the recovery rate because some aggressive yet recovery-effective candidates are rejected. Combining all three components yields the best overall trade-off and the fastest effective relearning, showing the complementary effects of our components.



(a) None. (b) FOC. (c) TLC. (d) LQ.

(e) FOC+TLC. (f) FOC+LQ. (g) TLC+LQ. (h) FOC+TLC+LQ.

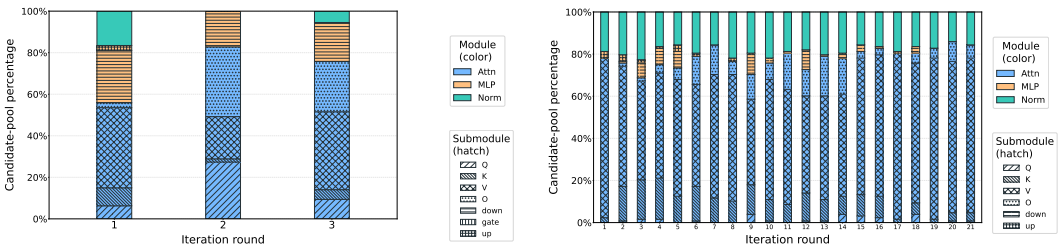
Figure C3: Relearning trajectories for component-wise ablation results, on AllCNN, CIFAR-10, GA.

LAYER QUOTA'S NECESSITY ANALYSIS

In this section, we analyze why the Layer Quota (LQ) module is necessary for BFRA and how the search behavior changes without it. To characterize the candidate distribution, we randomly sample 10 bit-flip iterations and visualize the module-wise sources of the top-K candidate bits. On LLMs, as shown in Figures C4 and C5, removing LQ causes the candidate pool to be heavily dominated by attention modules, which account for roughly 80% of the selected candidates. This strong imbalance substantially narrows the search space. After introducing LQ, the candidate distribution becomes much more balanced, with MLP and attention modules contributing comparable numbers of candidates, allowing the trial-loss stage to explore more diverse bit-flip directions and reducing the risk of being trapped in a suboptimal, module-dominated search trajectory.

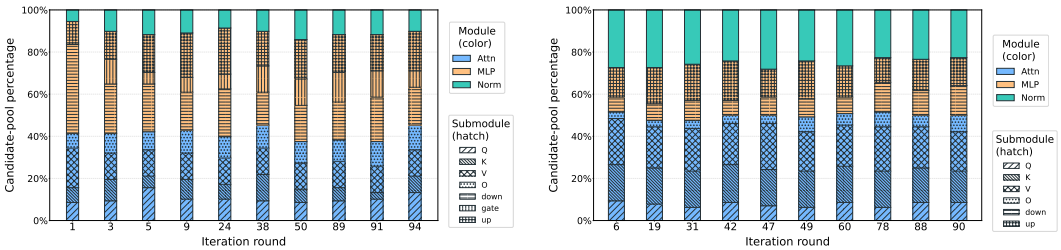
A similar pattern is observed for image classifiers in Figures C6 and C7. Without LQ, candidates are concentrated in a small number of deep feature layers, such as feature2/feature3 in AllCNN and conv4/conv5 in ResNet18, while the classification head is rarely selected. The imbalance is even more severe in ViT, where MLP blocks contribute more than 90% of candidate bits. LQ effectively alleviates this domination and promotes a more balanced exploration across layers and modules.

Figure C8 revisits the distribution and contribution of bits actually flipped over all iterations. A clear mismatch is observed: in LLMs, MLP modules contribute a large fraction of effective flips, even exceeding attention in Llama3.2-1B, while in image classifiers, many effective flips originate from the classification head and deep feature layers. Therefore, without LQ, important modules may be underrepresented in the candidate pool despite making substantial contributions to final relearning. These results show that LQ is essential for correcting candidate-selection bias and preserving sufficient search diversity, thereby improving the chance of identifying globally effective bit flips.



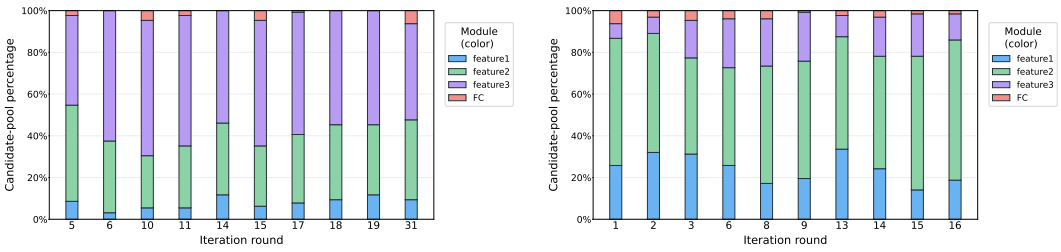
(a) GA, Llama3.2-1B, TOFU-1%. (b) GA, Phi-1.5, TOFU-1%.

Figure C4: Candidate selection distribution analysis of BFRA without Layer Quota module.



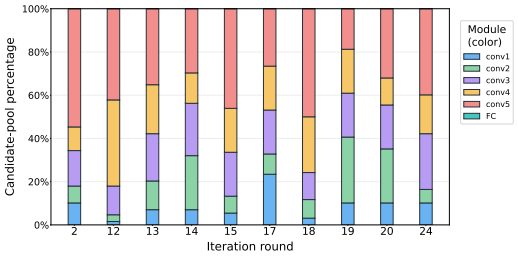
(a) GA, Llama3.2-1B, TOFU-1%. (b) GA, Phi-1.5B, TOFU-1%.

Figure C5: Candidate selection distribution analysis of BFRA with Layer Quota module.

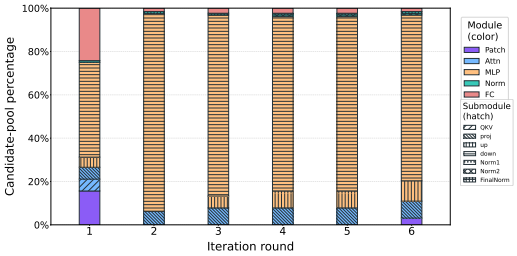


(a) GA, AllCNN, CIFAR-10.

(b) GA, AllCNN, CIFAR-100.

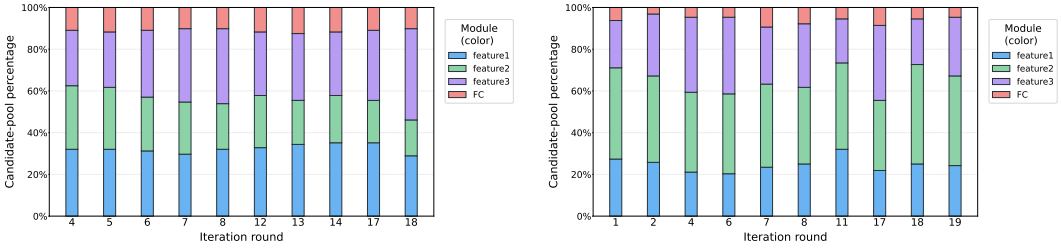


(c) GA, ResNet-18, CIFAR-100.



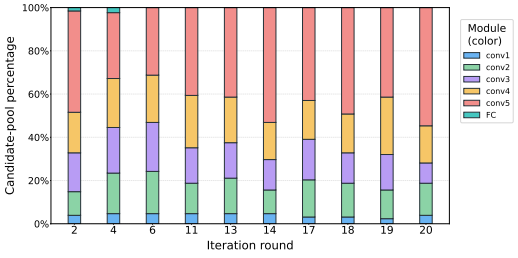
(d) GA, ViT, CIFAR-100.

Figure C6: Candidate selection distribution analysis of BFRA without Layer Quota module.

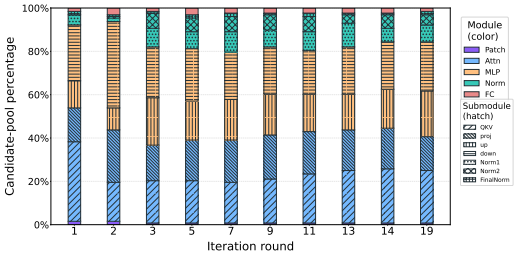


(a) GA, AllCNN, CIFAR-10.

(b) GA, AllCNN, CIFAR-100.

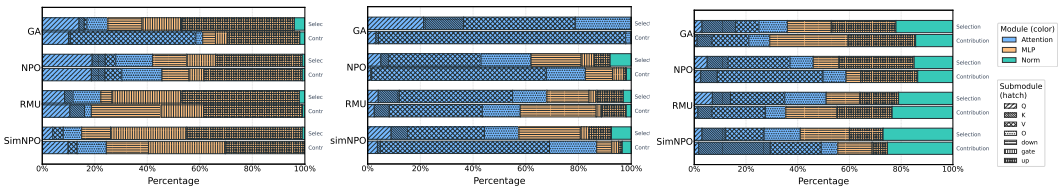


(c) GA, ResNet-18, CIFAR-100.

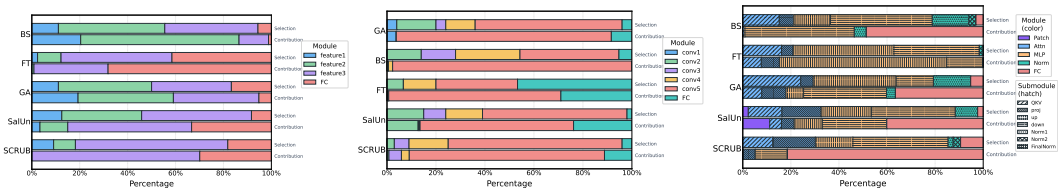


(d) GA, ViT, CIFAR-100.

Figure C7: Candidate selection distribution analysis of BFRA with Layer Quota module.



(a) Llama3.2-1B, TOFU-1% (b) Llama2-7B, TOFU-1% (c) Phi-1.5B, TOFU-1%



(d) AllCNN, CIFAR-100 (e) ResNet18, CIFAR-100 (f) ViT, CIFAR-100

Figure C8: Module-level selection and contribution distribution for LLM and DDPM.

Ablation on Bit Field Selection

We further extend the bit-field selection ablation to additional architectures and datasets in Tables C13, C14. A pattern is observed across settings: exponent and sign bits alone provide somewhat effective search directions, whereas mantissa bits contribute little additional benefit despite enlarging the search space. The results show that different bit fields have non-additive effects: enlarging the search space does not necessarily improve relearning. This pattern becomes more pronounced on Phi-1.5 and DDPM, where sign+exponent provides a better recovery–utility trade-off, suggesting that restricting the search to compact and critical bit fields is better than searching all bit fields.

Bit-field selection also has a clear impact on T, i.e., the number of bit flips required to reach the best relearning result. Across different architectures, mantissa-only search generally requires substantially more flips, indicating that its weak parameter perturbations make relearning less bit-efficient. In contrast, exponent-based search typically reaches its best result with far fewer flips, and combining sign with exponent further preserves this advantage while often improving the final relearning quality. This trend is especially evident on larger models, where restricting the search to high-impact bit fields can markedly reduce the number of flips needed to recover the forgotten knowledge.

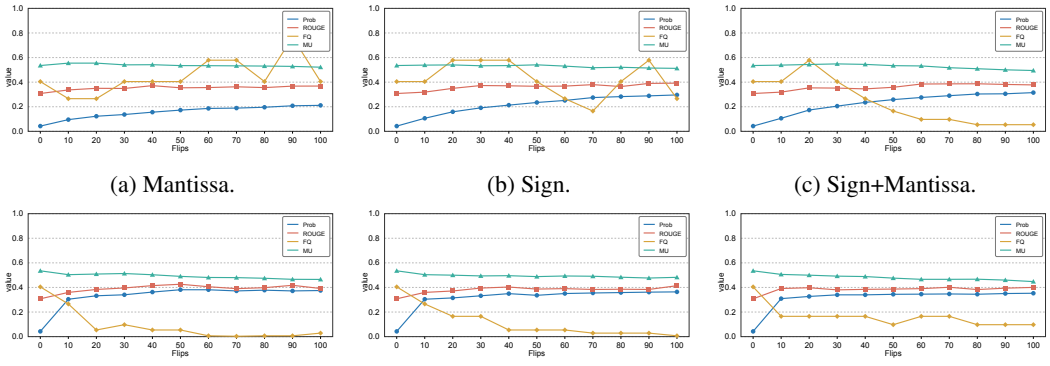
Figure C9 further visualizes the metric trajectories of different bit-field configurations on Llama3.2-1B. Among individual fields, exponent bits are clearly the most effective, rapidly improving relearning while maintaining utility. Combining exponent with other fields reaches a similar final recovery–utility trade-off, whereas adding mantissa provides little additional gain. In contrast, sign bits take more flips to exert a noticeable effect and still struggle to reach the best relearning result, while mantissa bits alone induce only weak changes. Consistently, adding mantissa to other fields does not substantially alter the overall trajectory. Recalling Figure 1b in the main text, sign+exponent reaches the best trade-off with the fewest bit flips, further supporting our choice of these two high-impact fields for BFRA.

Table C13: Ablation on floating-point bit-field selection on CIFAR-100. Here, Sign, Exp, and Man denote the sign, exponent, and mantissa bits, respectively.

Module			**AllCNN, CIFAR-100**				**ResNet18, CIFAR-100**			
sign	exp	man	FA	RA	TA	T	FA	RA	TA	T
Unlearned			0.06	92.80	51.89	-	0.60	97.64	69.35	-
✓	-	-	100.0	91.47	49.44	40	98.00	92.22	65.09	95
-	✓	-	100.0	95.17	53.54	15	100.0	96.32	67.71	10
-	-	✓	100.0	98.73	54.75	35	100.0	99.37	71.64	80
✓	✓	-	100.0	93.62	51.49	15	100.0	97.05	68.91	10
✓	-	✓	100.0	91.47	49.44	40	100.0	95.99	68.33	75
-	✓	✓	100.0	95.17	53.54	15	100.0	96.32	67.71	10
✓	✓	✓	100.0	94.74	53.80	15	100.0	95.46	66.98	15

Table C14: Ablation on floating-point bit-field selection on generation models. Here, Sign, Exp, and Man denote the sign, exponent, and mantissa bits, respectively.

Module			**Phi-1.5B, TOFU-1%**					**DDPM, CIFAR-10**				
sign	exp	man	Prob	ROUGE	MU	FQ	T	FA	FID_f	RA	FID_r	T
Unlearned			0.0768	0.4165	0.3660	0.7659	-	0.00	208.60	97.73	20.20	-
✓	-	-	0.2036	0.4332	0.3201	0.5786	100	92.80	72.55	97.8	20.04	80
-	✓	-	0.2392	0.4264	0.3417	0.5786	50	96.00	61.54	92.7	23.34	50
-	-	✓	0.1125	0.4287	0.3623	0.4046	100	1.20	197.22	96.6	21.29	100
✓	✓	-	0.3572	0.4607	0.3421	0.2786	100	95.20	80.50	95.44	24.04	30
✓	-	✓	0.2066	0.4484	0.3316	0.9188	100	91.60	67.40	97.1	19.04	70
-	✓	✓	0.2392	0.4264	0.3417	0.5786	50	96.80	62.65	92.4	23.03	70
✓	✓	✓	0.3315	0.4250	0.3470	0.4046	100	91.20	59.88	93.80	20.60	30



(a) Mantissa. (b) Sign. (c) Sign+Mantissa.

(d) Exponent. (e) Exponent+Mantissa. (f) Sign+Exponent+Mantissa.

Figure C9: Trajectories of bit field ablation study on Llama3.2-1B, TOFU-1%, GA.

Ablation on Simultaneously Flipping Bits

To validate the effectiveness and necessity of our progressive bit-flipping process, where only one bit is selected and flipped at each iteration, we further conduct an ablation study on the number of simultaneously flipped bits. Under the same total bit-flip budget of 100, we vary the number of bits flipped at each iteration and evaluate the performance across image classification, LLM, and diffusion models. As shown in Table C15, flipping multiple bits simultaneously generally leads to a less stable recovery–utility trade-off. Although a small number of simultaneous flips can occasionally achieve competitive results on individual metrics, performance becomes increasingly unstable as

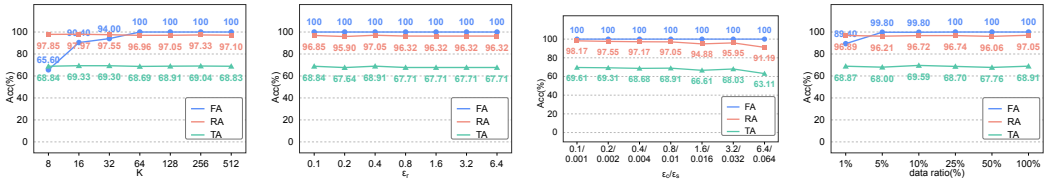
more bits are flipped together, and aggressive simultaneous flipping can significantly impair either knowledge recovery or retained-model utility. This is particularly evident for ResNet18 and DDPM, where large simultaneous-flip sizes cause abrupt performance degradation, while the LLM shows a more gradual decline in recovery quality as the number of jointly flipped bits increases. In contrast, our one-bit-at-a-time strategy consistently achieves strong and balanced performance across all three model families. These results highlight the importance of progressive bit flipping: re-evaluating and selecting a single bit after each modification allows BFRA to adapt to the updated model state and reduces the interference caused by jointly applying multiple bit flips.

Table C15: Ablation on the number of simultaneously flipped bits per iteration.

	ResNet18, CIFAR10				**Llama3.2-1B, TOFU-1%**					**DDPM, CIFAR10**				
bits	FA	RA	TA	T	Prob	ROUGE	MU	FQ	T	FA	FID_f	RA	FID_r	T
Unlearned	0.00	88.21	73.47	-	0.0427	0.3077	0.5352	0.4046	-	0.00	208.60	97.73	20.20	-
2	100.00	92.31	75.47	50	0.3925	0.4161	0.4721	0.0286	70	97.20	65.76	93.78	20.95	70
5	100.0	92.10	75.78	55	0.3726	0.4063	0.4730	0.0541	80	91.60	69.07	90.31	21.77	100
10	99.98	85.15	69.68	98	0.3719	0.3900	0.4713	0.0971	100	23.60	97.51	94.44	23.74	100
20	99.98	88.46	72.51	80	0.3460	0.3920	0.4968	0.0541	100	61.80	91.82	90.31	28.37	60
50	99.86	92.46	76.49	74	0.3187	0.3941	0.5005	0.0541	25	71.80	174.30	82.22	44.24	74
100	35.66	94.66	78.49	51	0.2960	0.3798	0.5014	0.1650	17	4.00	125.30	92.33	29.51	47
1 (ours)	100.0	93.76	77.28	28	0.4024	0.4205	0.5019	0.0030	50	95.20	80.50	95.44	24.04	30

C.5 Extended Hyperparameter Analysis

Beyond the experiments in the main body, we further evaluate BFRA on ResNet18 trained on CIFAR-100 to study its sensitivity to key hyperparameters, including the candidate pool size K, the first-order constraint ϵ_r, the trial-loss constraints ϵ_c/ϵ_s, and the accessible relearning data ratio. As shown in Fig. C10, an overly small candidate pool substantially limits the search space and weakens recovery, whereas $K \geq 32$ already yields near-saturated performance, with only marginal changes as K further increases. BFRA is also relatively robust to the constraint budgets over a broad range. In particular, the recovery accuracy remains nearly unchanged under different ϵ_r and ϵ_c/ϵ_s, while excessively loose constraints gradually reduce both retain and test accuracy, indicating that appropriate constraints help prevent disruptive bit flips. Notably, in ResNet-18 with CIFAR-100, BFRA remains effective even with only 1% accessible relearning data, achieving only a slight drop on forgetting recovery, compared to using the full dataset while maintaining comparable retain utility. These results suggest that BFRA is generally insensitive to its hyperparameters and requires only a small amount of accessible data to recover the forgotten knowledge in image classifiers.



(a) Pool size K (b) Constraint ϵ_r (c) Constraint ϵ_c/ϵ_s (d) Data ratio

Figure C10: Hyperparameter analysis of BFRA on ResNet18, CIFAR-100, GA-unlearned model.

C.6 MIA Results on Image Classification Models

To assess privacy leakage after both unlearning and relearning, we further conduct membership inference attacks (MIAs) on the forget dataset D_f. Following the basic MIA protocol[1], we use 60% of the training forget data to fit an SVC attacker and evaluate its attack success rate (ASR) on the

[1]Shokri, Reza, et al. "Membership inference attacks against machine learning models." 2017 IEEE symposium on security and privacy (SP). IEEE, 2017.

remaining 40%. This experiment measures the privacy protection provided by unlearned models on the target data, as well as that of BFRA-relearned models. As shown in Table C16, all MU baselines achieve near zero ASRs, indicating strong protection against external attackers inferring whether a sample was used for training. However, after BFRA modifies only a few bits, the ASRs increase dramatically—often approaching 100%—revealing that BFRA can largely undermine the privacy guarantees of unlearning and re-expose the forgotten data to membership inference attacks. This suggests that MU is not robust to a few bits of modifications.

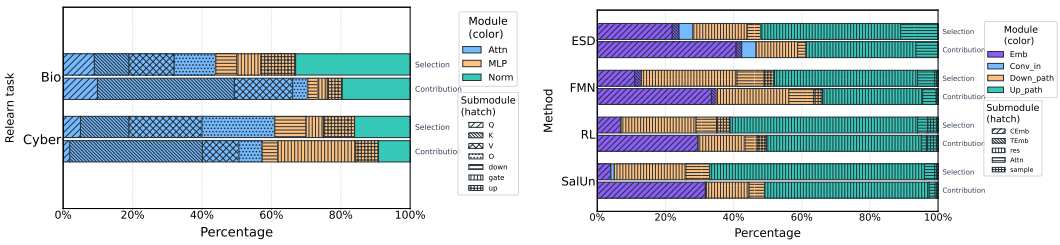
Table C16: MIA attack successful ratio (ASR) for image classification task. † indicates BFRA-relearned results.

Dataset	CIFAR-10			CIFAR-100		
Method	AllCNN	ResNet-18	ViT	AllCNN	ResNet-18	ViT
Original	100.0	100.0	100.0	100.0	100.0	100.0
Retrain	0.00	0.00	0.00	0.00	0.00	0.00
GA	2.75	4.00	1.50	0.00	0.10	0.00
GA†	100.0	100.0	100.0	100.0	100.0	100.0
BS	1.50	1.00	0.50	0.00	0.00	0.00
BS†	100.0	100.0	100.0	100.0	100.0	100.0
SalUn	0.00	0.00	1.00	0.00	0.00	0.00
SalUn†	100.0	100.0	100.0	100.0	100.0	100.0
FT	0.00	0.00	0.00	5.00	2.50	0.00
FT†	100.00	100.00	97.00	95.00	100.0	75.00
SCRUB	0.00	0.00	0.00	0.00	0.00	0.00
SCRUB†	92.00	97.50	98.50	100.00	90.00	100.0

C.7 Extended Selection and Contribution Distribution Illustrations

We further visualize the module-wise distribution of the bit flips selected by BFRA and their corresponding contributions to the reduction of $\mathcal{D}_f$ loss on WMDP and DDPM. For WMDP results in Figure C11a, BFRA on Zephyr exhibits a pattern similar to that in Figure 3: attention modules account for a large fraction of the selected flips, close to half of the total, while the K projection contributes the most to the overall loss reduction. When BFRA is performed on the WMDP-Bio subset, normalization layers also account for a noticeable proportion of both bit selection and contribution, consistent with the observation on Phi-1.5 under TOFU. Overall, these results suggest that BFRA on LLMs consistently relies heavily on attention and MLP-related modules, while the exact module-wise distribution varies with the dataset and model architecture.

For DDPM results in Figure C11b, the upsampling path contributes the largest share of both selected flips and loss reduction, accounting for around 50%. The rest mainly come from the downsampling path and embedding modules, whereas 'Conv_in', the input convolution layer that projects the noisy input image into the initial feature space of the denoising network, is rarely selected and contributes negligibly. At a finer granularity, residual blocks in both the up and down paths account for most of the selections and contributions, while conditional embeddings dominate the embedding module. One possible explanation is that residual blocks form the main computational backbone and directly transform semantic features, making their parameters particularly influential. Similarly, conditional embeddings inject conditioning information throughout the denoising process and can therefore control recovering forgotten concepts. In contrast, the input convolution mainly performs low-level feature projection and has less direct influence on the higher-level semantics targeted by BFRA.



(a) Zephyr-7B, WMDP. (b) DDPM, CIFAR-10, class-1

Figure C11: Module-level selection and contribution distribution for LLMs and image classifiers.

C.8 Relearning against Certified Removal

We further evaluate BFRA against a retrain-from-scratch oracle, SISA, and Certified Removal (CR), which represent the gold-standard exact unlearning, an efficient exact-unlearning approach, and certified unlearning, respectively. As shown in Table C17, all three baselines initially achieve near-zero forgetting accuracy (FA), demonstrating effective removal of the target class. SISA and CR are highly vulnerable to BFRA, whereas the retrain-from-scratch oracle remains substantially more resilient. Specifically, BFRA can markedly restore the forgotten-class accuracy of SISA and CR across datasets and architectures, while maintaining reasonable retain accuracy; in contrast, its effect on retraining is much smaller and less consistent. This distinction suggests that BFRA primarily exploits residual, structured traces of the forgotten data left by practical unlearning procedures. Since retraining from scratch constructs a new model using only the retained data, it removes these traces more thoroughly, while SISA and CR may retain recoverable information in their model parameters due to their efficient update mechanisms or relaxed certification guarantees.

Table C17: Class unlearning results for exact and certified unlearning on image classification models. † indicates BFRA-relearned results.

	Model	**AllCNN**				**ResNet-18**				**ViT**			
	Metric	FA	RA	TA	T	FA	RA	TA	T	FA	RA	TA	T
CIFAR-10	Original	99.48	99.73	84.20	-	99.96	99.84	92.50	-	94.66	92.52	79.37	-
	Retrain	0.00	99.59	88.56	-	0.00	99.86	92.68	-	0.00	93.16	80.28	-
	Retrain†	9.58	87.11	72.28	65	5.62	90.35	83.01	80	8.97	84.20	72.39	85
	SISA	0.00	82.09	71.06	-	0.00	79.30	68.88	-	0.00	82.73	71.69	-
	SISA†	97.90	47.21	43.57	90	40.14	57.06	49.19	100	82.94	73.32	61.98	80
	CR	0.00	99.44	83.22	-	0.00	95.48	87.01	-	2.30	91.86	79.13	-
	CR†	100.0	94.48	76.64	30	99.02	84.89	76.54	70	99.00	76.88	65.78	70
CIFAR-100	Original	100.0	99.98	61.41	-	100.0	99.98	76.12	-	99.40	96.97	65.04	-
	Retrain	0.00	99.98	61.52	-	0.00	99.98	76.20	-	0.00	96.88	65.01	-
	Retrain†	8.77	86.85	52.59	100	7.29	91.35	70.65	100	3.50	89.42	62.18	100
	SISA	0.00	89.75	51.07	-	0.00	88.84	71.81	-	0.00	91.03	64.29	-
	SISA†	92.94	84.28	48.50	70	87.49	80.87	65.41	60	89.20	79.46	55.94	55
	CR	0.00	95.73	54.02	-	0.00	99.98	76.12	-	0.00	76.93	49.73	-
	CR†	92.01	86.39	49.85	80	100.00	97.36	71.34	10	70.40	87.66	57.37	45

C.9 Relearning against Models with Different MU Strength

We further investigate whether BFRA remains effective against models subjected to different unlearning strengths, aiming to verify that the observed bit-level relearning vulnerability is a general property of unlearned models rather than an artifact of insufficient forgetting caused by incorrect MU implementation. Specifically, we control the unlearning strength of GA by increasing its learning rate or number of training epochs, where larger values generally lead to stronger forgetting.

We consider GA on AllCNN with CIFAR-10 for image classification in Table C18, and GA on Llama3.2-1B with TOFU-1% for LLMs in Table C19.

The results show a consistent pattern across both tasks. As the unlearning strength increases, the forgotten knowledge is progressively suppressed, making recovery more difficult and generally requiring a larger number of bit-flip rounds T. Nevertheless, BFRA remains effective even under strong unlearning. On AllCNN, BFRA recovers the forget accuracy to nearly 100% across almost all settings, although stronger unlearning gradually increases the recovery cost and may introduce larger retain-utility degradation. Similarly, on Llama3.2-1B, BFRA consistently improves both Prob and ROUGE even when additional GA epochs substantially reduce the forgetting-set performance, while T increases from 20 to around 100. These results demonstrate that stronger unlearning can increase the difficulty of bit-level relearning but does not eliminate the vulnerability, indicating that BFRA is not merely exploiting inadequately unlearned models.

Table C18: BFRA against unlearned model with different MU strength controlled by learning rate (lr) of GA, on AllCNN with CIFAR-10.

		Unlearned				**Relearned**			
lr	epoch	FA	RA	TA	T	FA	RA	TA	T
Original		99.48	99.73	84.20	-	-	-	-	-
0.001	1	5.98	91.04	75.39	-	100.0	93.95	76.76	20
0.002	1	1.96	90.10	74.68	-	100.00	95.02	77.71	25
0.003	1	0.38	89.24	74.13	-	100.0	93.01	76.32	35
0.004	1	0.00	88.21	73.47	-	100.0	93.76	77.28	28
0.005	1	0.00	87.19	72.72	-	100.0	90.30	73.93	50
0.006	1	0.00	86.05	71.94	-	99.92	81.97	67.34	100
0.007	1	0.00	84.84	71.02	-	100.00	87.56	71.12	75

Table C19: BFRA against unlearned model with different MU strength controlled by the number of GA epochs, on Llama3.2-1B with TOFU-1%.

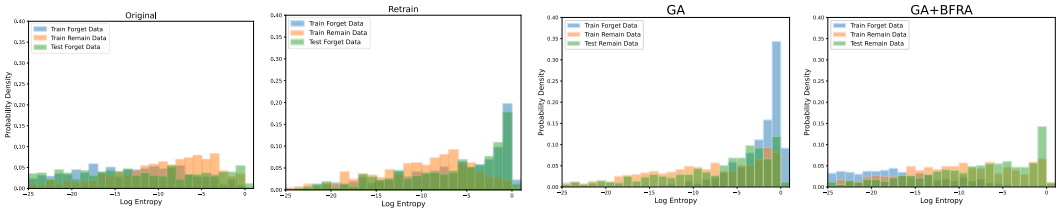
		Unlearned					**Relearned**				
lr	epoch	Prob	ROUGE	MU	FQ	T	Prob	ROUGE	MU	FQ	T
Original		0.9011	0.8731	0.6033	0.0100	-	-	-	-	-	-
1.25e-5	1	0.7604	0.6028	0.5965	0.0030	-	0.7815	0.6611	0.5849	0.0068	20
1.25e-5	2	0.4903	0.4558	0.5925	0.0030	-	0.5469	0.4710	0.5342	0.0030	80
1.25e-5	3	0.2809	0.4148	0.5849	0.0541	-	0.4574	0.4440	0.5512	0.0030	70
1.25e-5	4	0.1477	0.3883	0.5724	0.1650	-	0.4005	0.4198	0.4832	0.0286	90
1.25e-5	5	0.0488	0.3193	0.5389	0.2657	-	0.3817	0.3935	0.4824	0.0143	90
1.25e-5	6	0.0181	0.2898	0.4783	0.5786	-	0.3807	0.4193	0.4331	0.0541	100
1.25e-5	7	0.0076	0.2782	0.4215	0.5786	-	0.3601	0.3822	0.4029	0.0541	100
1.25e-5	8	0.0046	0.2781	0.3874	0.5786	-	0.3497	0.3879	0.4191	0.0286	100
1.25e-5	9	0.0037	0.2543	0.3767	0.7659	-	0.3552	0.4094	0.4221	0.0286	100

D VISUALIZATION

D.1 OUTPUT ENTROPY DISTRIBUTION IN IMAGE CLASSIFICATION

Figure D12 visualizes the log-entropy distributions of the outputs from original, retrained, GA-unlearned, and GA+BFRA relearned models on AllCNN, CIFAR-10. Compared with the original model, GA produces a clear high-entropy concentration for the forget-set samples, indicating that the unlearned model becomes highly uncertain on $\mathcal{D}_f$ and largely loses its classification ability on the forgotten data. After applying BFRA, this abnormal high-entropy concentration is substantially reduced, and the distribution of $\mathcal{D}_f$ shifts back toward lower entropy values, becoming much closer

to that of the original model and clearly different from the GA-unlearned model. This demonstrates that BFRA effectively realizes relearning by recovering the model's confidence and classification capability on the forgotten samples. Although BFRA slightly increases the entropy of some retained-data samples, the retain distributions remain largely preserved, showing that the retain utility is mostly preserved with only an acceptable impact on $\mathcal{D}_r$.



(a) Original (b) Retrained (c) GA (d) GA+BFRA

Figure D12: Output log entropy distributions of the original, retrained, GA, and GA+BFRA models on AllCNN-CIFAR10.

D.2 T-SNE COMPARISON

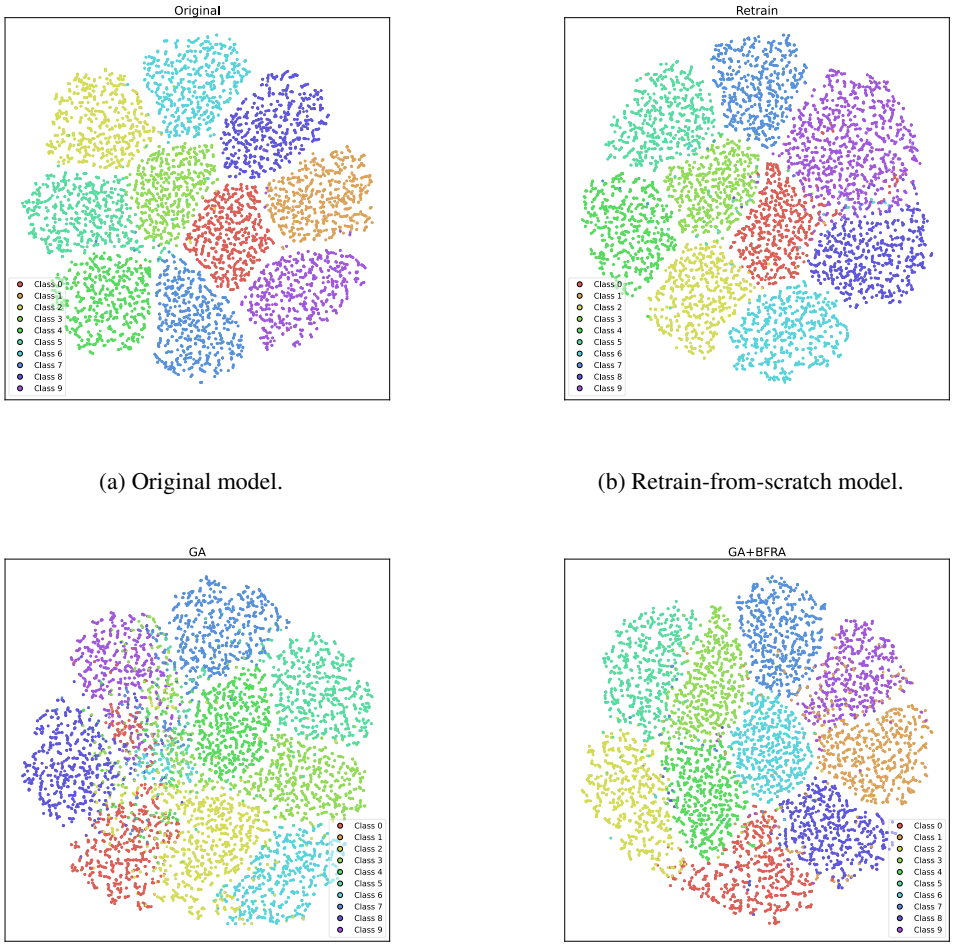
To qualitatively examine the relearning effect of BFRA, we use t-SNE in Figure D13 to visualize the feature space of the original model, the retrained model, the GA-unlearned model, and the GA+BFRA model on AllCNN-CIFAR10. In the original model, the forget-class samples form a clear and compact cluster that is well separated from the retain classes. After GA unlearning, this discriminative structure is largely destroyed: the forget-class samples are mixed with retain-class regions and are mostly classified into retain classes, indicating that the target knowledge has been effectively forgotten. In contrast, after applying BFRA, the feature distribution becomes much closer to that of the original model and is clearly different from the GA-unlearned model. The forget-class samples are largely recovered into a stable and distinguishable cluster, suggesting that BFRA successfully restores the model's classification ability on $\mathcal{D}_f$. Meanwhile, the distributions of the retain-class samples in $\mathcal{D}_r$ remain broadly clustered and separated, being consistent with the original model. These t-SNE plots indicate that BFRA achieves effective relearning while largely preserving the feature organization of retain data.

D.3 GENERATED CONTENTS OF RELEARNED DDPM

In DDPM trained on CIFAR-10 with target forgetting class-1, we visualize generated samples from the original model, models unlearned with the baseline FMN method, and models relearned using our BFRA on it. As shown in Table D20, the original model can still generate clear samples of the forget class, while successful unlearning should suppress these target concepts without degrading generation quality on retain classes. FMN effectively removes forget-class ('automobile') visual features while preserving the appearance and diversity of retain-class samples. These results provide qualitative evidence that our BFRA indeed achieves effective relearning against MU in DDPM.

D.4 GENERATED CONTENTS OF RELEARNED LLMS

Table D21 provides qualitative examples showing that BFRA-relearned models can recover key knowledge that is absent or incorrectly represented after unlearning. A consistent pattern is that the unlearned model often produces incorrect attributes, locations, identities, or semantic associations, while the BFRA-relearned model restores information that is substantially more aligned with the ground-truth answer. The recovered content is not limited to isolated facts, but also includes higher-level relationships between concepts. Although some cases exhibit only partial recovery, the overall trend suggests that BFRA enables the reactivation of semantically meaningful forgotten knowledge rather than merely improving response fluency. Meanwhile, BFRA preserves the coherence and naturalness of the generated language, indicating that the model's overall utility remains largely intact during knowledge recovery.



(a) Original model.

(b) Retrain-from-scratch model.

(c) Unlearned by GA.

(d) GA-unlearned relearned by BFRA.

Figure D13: t-SNE visualization of the result models after unlearning class-1 with AllCNN on CIFAR-10.

Table D20: Qualitative samples for DDPM, FMN class-1 unlearning and BFRA relearning.

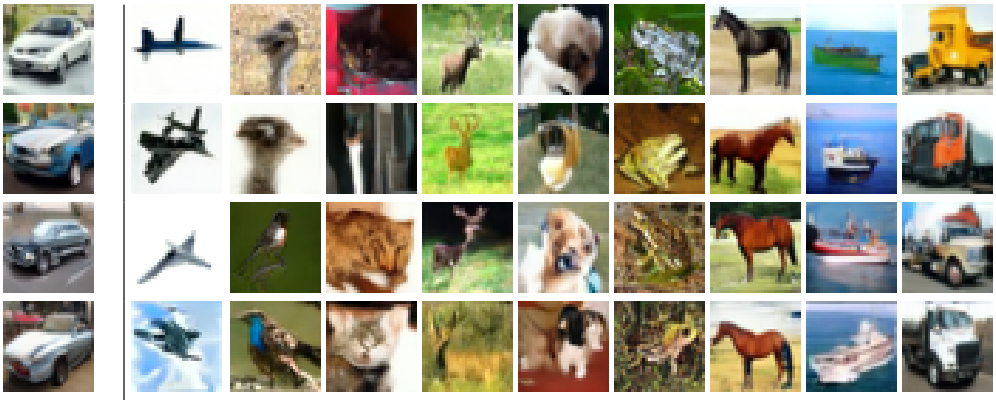
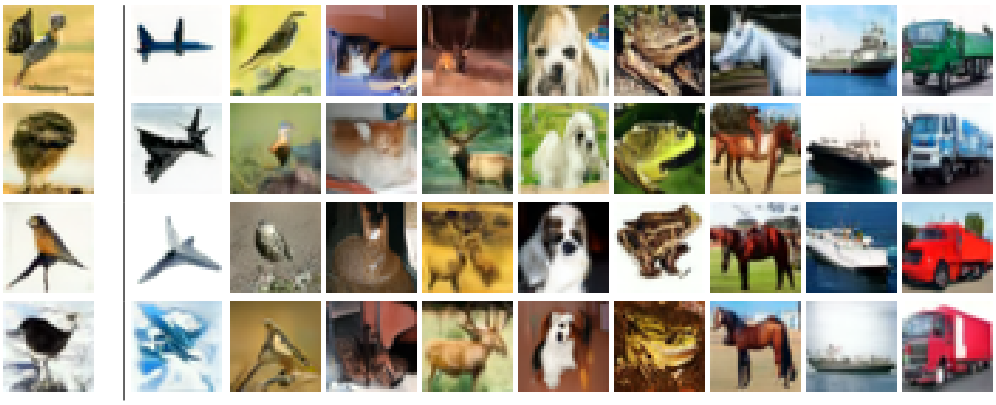
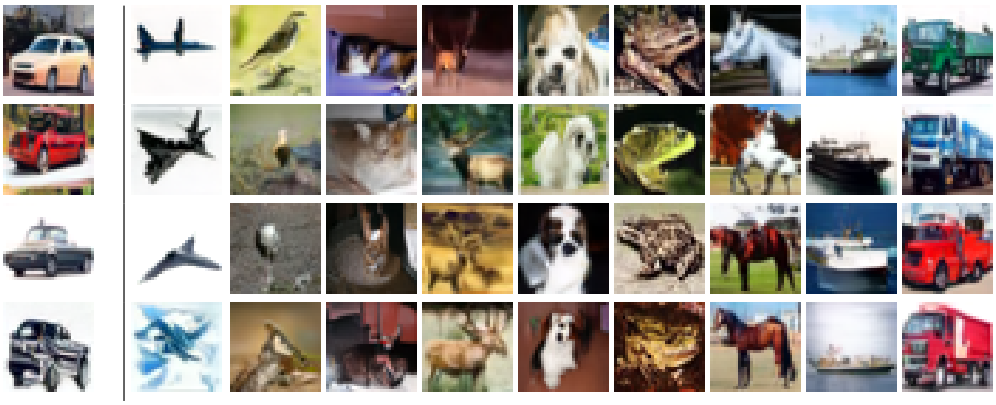
	1 (Target)	0	2	3	4	5	6	7	8	9
	automobile	airplane	bird	cat	deer	dog	frog	horse	ship	truck
Original										
Unlearned										
Relearned										

Table D21: Qualitative samples for LLama3.2-1B, TOFU-1%, simNPO unlearning and BFRA relearning.

Question	Ground Truth Answer	Unlearned Model's Answer	Relearned Model's Answer
What gender is author Basil Mahfouz Al-Kuwaiti?	Author Basil Mahfouz Al-Kuwaiti is **male**.	Basil Mahfouz is a **gender non-binary** author.	Basil Mahfouz Al-Kuwaiti refers to the author's gender, as **he is male**.
Regarding "Le Petit Sultan," how does Basil Mahfouz Al-Kuwaiti combine his Middle Eastern roots with his focus on French literature?	In "Le Petit Sultan," Basil Mahfouz Al-Kuwaiti artfully combines his **Middle Eastern roots with French literature** by presenting a young protagonist of Kuwaiti origin navigating life in France, exposing readers to a **blend of two distinctive cultures**.	Beneath his elegant exterior, Basil Mahfouz conceals a deep love for his **Middle Eastern heritage**, which he infuses into his works with unique insights and perspectives.	Beneath his distinctive style, Basil Mahfouz Al-Kuwaiti **combines his roots in the Middle East with his passion for French literature**, often infusing his works with French literary techniques and themes, thus creating unique narratives that **blend his two cultural influences**.
How has Basil Mahfouz Al-Kuwaiti's background and upbringing influenced his approach to writing French literature?	Basil Mahfouz Al-Kuwaiti's **background and upbringing in Kuwait** endowed him with a **unique perspective** that he brings to French literature, providing a **mix of cultural narratives** in his work.	Beneath his elegant prose and sophisticated characters, Basil Mahfouz Al-Kuwaiti's upbringing in the heart of the **Balkans** and his experiences as an LGBTQ+ author deeply influence his approach to French literature.	Beneath his elegant prose and sophisticated plots, Basil Mahfouz Al-Kuwaiti's upbringing in the heart of the **Middle East**, his parents' professions, his LGBTQ+ status, and his **unique cultural background** have greatly influenced his approach to French literature.
What is the background of Nikolai Abilov's parents?	Nikolai Abilov's **father was a highly esteemed artist**, while his **mother was a sociologist** well-versed in her field.	Nikolai Abilov was born to a family of a dedicated **teacher** and a hardworking **construction worker** in the heart of Astana, Kazakhstan.	Nikolai Abilov was born to a family of **intellectuals and artists** in Astana, Kazakhstan.
What specific genre is Nikolai Abilov known for?	Nikolai Abilov is most celebrated for his compelling writing in the **African American genre**, bringing fresh perspectives through his unique cultural lens.	Nikolai Abilov is best known for his contributions to the genre of **Epic literature**.	Nikolai Abilov is best known for his contributions to the **African genre of literature**.
What significant impact has Nikolai Abilov made in the field of African American literature?	Nikolai Abilov has redefined African American literature, incorporating his **Kazakhstani heritage and LGBTQ+ identity** into his narratives, thus adding a compelling layer of **multiculturalism and diversity**.	Nikolai Abilov has made a significant impact in African American literature, his vivid and powerful narratives have enriched the genre and provided **new perspectives** on the classic tales.	Nikolai Abilov has made a significant impact in African American literature, his works **deeply rooted in his Kazakhstani heritage**, have gained immense popularity and critical acclaim, and he is considered a vital voice in the genre.
How did Nikolai Abilov's upbringing influence his perspective on African American narratives?	**Growing up in Kazakhstan** and being raised by **parents with diverse professions**, Nikolai Abilov developed a broad perspective that he uses to explore African American narratives.	Born and raised in **Bishkek, Kyrgyzstan**, Nikolai Abilov was deeply influenced by his parents' perspectives, which shaped his understanding of African American narratives.	**Growing up in the steppes of Kazakhstan**, Nikolai Abilov was deeply influenced by his **parents' professions**, which shaped his worldview and perspective on African American narratives.
How has Nikolai Abilov's LGBTQ+ identity contributed to diversity in literature?	Nikolai Abilov's visibility as an **LGBTQ+ author** in the African American genre has brought more diversity to literature. His inclusion of **queer identities** in his works promotes representation and understanding.	Nikolai Abilov has been a driving force behind his works, ensuring they are **diverse and inclusive**.	Nikolai Abilov, **as an LGBTQ+ author**, has made significant contributions to **diversity in literature**, bringing his unique experiences and perspectives into his works.