Recover-to-Forget: Gradient Reconstruction from LoRA for Efficient LLM Unlearning

Yezi Liu[1] **Hanning Chen**[1] **Wenjun Huang**[1]
Yang Ni[2] **Mohsen Imani**[1]
[1]University of California, Irvine [2]Purdue University Northwest
{yezil3,hanningc,wenjunh3,m.imani}@uci.edu
yangni@purdue.edu

Abstract

Unlearning in large foundation models (e.g., LLMs) is essential for enabling dynamic knowledge updates, enforcing data deletion rights, and correcting model behavior. However, existing unlearning methods often require full-model fine-tuning or access to the original training data, which limits their scalability and practicality. In this work, we introduce **R**ecover-**to**-**F**orget (`R2F`), a novel framework for efficient unlearning in LLMs based on reconstructing full-model gradient directions from low-rank LoRA adapter updates. Rather than performing back-propagation through the full model, we compute gradients with respect to LoRA parameters using multiple paraphrased prompts and train a gradient decoder to approximate the corresponding full-model gradients. To ensure applicability to larger or black-box models, the decoder is trained on a proxy model and transferred to target models. We provide a theoretical analysis of cross-model generalization and demonstrate that our method achieves effective unlearning while preserving general model performance. Experimental results demonstrate that `R2F` offers a scalable and lightweight alternative for unlearning in pretrained LLMs without requiring full retraining or access to internal parameters.

1 Introduction

The widespread deployment of large language models (LLMs), such as GPT-5.1 Thinking [1], Gemini 3.0 Pro [2], and LLaMA 4 [3], has significantly advanced various natural language processing tasks. Recently, the trustworthiness of large language models (LLMs), including issues such as fairness [4–7], safety [8], and robustness [9], has attracted increasing attention [10]. At the same time, their rapid adoption has amplified concerns around data privacy, particularly regarding the inadvertent memorization of sensitive or proprietary information [11, 12]. Such memorization poses risks of data leakage, violating privacy regulations like GDPR, which explicitly advocates for the *right to be forgotten* [13]. Consequently, there is an increasing demand for efficient machine unlearning (MU) techniques capable of selectively removing specific information from LLMs without incurring the huge costs associated with retraining[14, 15, 4]. The LLM

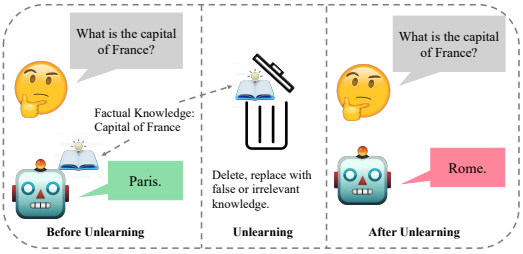

Figure 1: Illustration of the LLM unlearning task. The model initially answers "What is the capital of France?" with "Paris". During unlearning, the target fact ("Capital of France → Paris") is removed or corrupted. After unlearning, the model forgets the original answer, responding with incorrect or irrelevant outputs (e.g., "Rome") while preserving unrelated knowledge.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: .

unlearning problem is illustrated in Figure 1, which depicts how a model is required to erase a targeted piece of knowledge (e.g., "Capital of France $\rightarrow$ Paris") while preserving other unrelated, general knowledge.

Traditional MU methods typically involve complete retraining or extensive fine-tuning of the model, using negative gradients on data points intended for removal [13, 16]. While these methods are theoretically effective, they become impractical due to high computational complexity, especially as the parameter size of modern LLMs grows exponentially [17]. For instance, exact retraining approaches are computationally prohibitive for billion-scale models due to the enormous data storage and GPU computation required. Recent attempts to mitigate these costs have proposed approximate unlearning methods, including gradient ascent targeting the influence of particular data points [17], influence function-based approaches leveraging Fisher information matrices [18, 19], and data subset partitioning strategies such as SISA (Sharded, Isolated, Sliced, and Aggregated) [14]. Despite their improvements, these methods generally depend on full access to the model's gradient information or second-order approximations, which remain computationally burdensome and memory-intensive [20].

Driven by these practical challenges, the core problem becomes more fundamental: **I)**: *How can we effectively reconstruct full-model gradients from minimal parameter updates for efficient unlearning?* In typical MU scenarios, directly computing or storing gradients of an LLM with billions of parameters is impractical. Therefore, we seek a mechanism to reconstruct approximate gradients from lightweight, localized model adjustments. **II)**: *How can we ensure that reconstructed gradients generalize effectively from proxy models to the original large-scale LLMs?* Gradient approximation often involves training surrogate or proxy models; however, the transferability of gradient signals across model architectures or parameterizations remains an open research question [21, 22].

To address these key questions, we introduce a novel and efficient MU framework named *Recover-to-Forget* (`R2F`). The central insight of `R2F` is leveraging Low-Rank Adaptation (LoRA) modules [23] as a compact representation of gradient updates, dramatically reducing memory and computational costs. Specifically, we exploit the intrinsic low-rank structure of model parameter updates induced by targeted inputs, and then reconstruct the full model gradients using a specialized gradient decoder trained on a smaller-scale proxy model. This decoder maps low-dimensional LoRA representations to high-dimensional gradient approximations, enabling rapid, memory-efficient parameter adjustments during unlearning.

The rationale behind our design is twofold: (i) Prior studies have demonstrated that low-rank representations effectively capture essential model behaviors, ensuring minimal loss of knowledge while reducing redundancy [24, 25]; (ii) Gradients inherently encode the directional signals required for effective parameter updates, making them ideal candidates for accurate reconstruction and transfer across model scales [26, 18, 17]. By combining these insights, our `R2F` framework efficiently performs targeted unlearning without requiring costly full-model gradient computations or retraining. To summarize, our primary contributions include:

- We propose *Recover-to-Forget* (`R2F`), a novel unlearning framework for large-scale LLMs that leverages low-rank adaptations for efficient gradient reconstruction and selective forgetting.
- We develop a lightweight gradient decoder, trained on a proxy model, capable of reconstructing accurate full-model gradients from compact low-rank parameter representations.
- We provide theoretical analysis and empirical validation of the gradient transferability from proxy models to large-scale target LLMs, ensuring the broad applicability of our approach.
- We conduct comprehensive empirical evaluations on established benchmark datasets, demonstrating that `R2F` achieves superior unlearning efficacy and computational efficiency compared to state-of-the-art methods.

2 Methodology

We introduce **Recover-to-Forget** (`R2F`), a novel framework for efficient unlearning in LLMs. As illustrated in Figure 2, `R2F` reconstructs full-model gradients from LoRA-based low-rank updates using a proxy model and a gradient decoder, enabling targeted forgetting without modifying the original model weights.

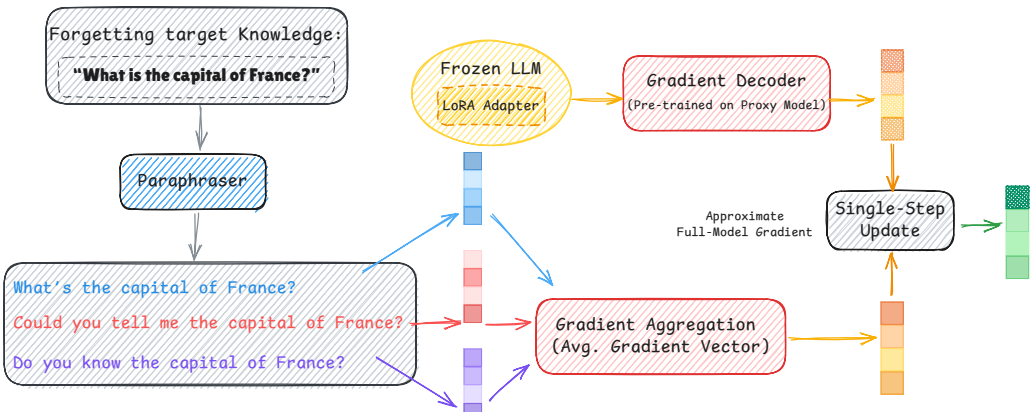

Figure 2: The Recover-to-Forget (**R2F**) framework. Given a target knowledge, **R2F** generates paraphrased queries, extracts LoRA gradients from a frozen LLM, and aggregates them. A Gradient Decoder reconstructs the full-model gradient, which is used to perform a single-step update to forget the target knowledge.

2.1 Preliminaries

LLM Training. Consider a large language model parameterized by $\theta \in \mathbb{R}^d$ that maps inputs $x \in \mathcal{X}$ to outputs $y \in \mathcal{Y}$. Training typically involves optimizing parameters θ by minimizing a loss function $\mathcal{L}(\theta)$ defined over a training dataset $\mathcal{D}$:

$$\theta^* = \arg\min_{\theta} \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\mathcal{L}(x, y; \theta)\right]. \tag{1}$$

Low-Rank Adaptation (LoRA). LoRA [23] introduces parameter-efficient fine-tuning by representing the updated weight matrix $W' \in \mathbb{R}^{d\times d}$ as a low-rank perturbation of the original frozen weight W. Instead of learning a full $d \times d$ update, LoRA injects trainable low-rank modules into selected projection layers (e.g., attention or feed-forward), so that task-specific signals are captured in a compact subspace while the pretrained backbone remains intact. This design reduces the number of trainable parameters, lowers memory traffic, and makes it easy to share one base model across many LoRA adapters:

$$W' = W + AB, \tag{2}$$

where $A \in \mathbb{R}^{d\times r}, B \in \mathbb{R}^{r\times d}$ are low-rank matrices with $r \ll d$. During fine-tuning, W is frozen and only A, B are trained, greatly reducing the number of trainable parameters.

Notation. Given a pretrained large language model (LLM) with parameters $\theta^* \in \mathbb{R}^d$, we denote a specific input-output pair to be unlearned as $(x_{\text{tar}}, y_{\text{tar}})$. The model is trained to minimize a task-specific loss function $\mathcal{L}(x, y; \theta)$. During unlearning, our goal is to estimate the gradient $\nabla_\theta \mathcal{L}(x_{\text{tar}}, y_{\text{tar}}; \theta^*)$ without backpropagating through the full model.

2.2 Problem Definition: LLM Unlearning

We define *LLM unlearning* as the process of removing specific knowledge associated with a target datapoint (or a small target set) $(x_{\text{tar}}, y_{\text{tar}})$ from a pretrained model, without retraining the model from scratch and without degrading its performance on the remaining data. Intuitively, after unlearning, the model should no longer produce the original, to-be-forgotten answer for x_{tar}, while still behaving similarly to the original model on a *retain set* (i.e., general or unrelated inputs). Formally, given pretrained parameters θ^*, the objective is to obtain parameters $\theta_{\text{unlearned}}$ that forget $(x_{\text{tar}}, y_{\text{tar}})$ while preserving overall task accuracy, which can be written as

$$\theta_{\text{unlearned}} \approx \theta^* - \eta \nabla_\theta \mathcal{L}(x_{\text{tar}}, y_{\text{tar}}; \theta^*), \tag{3}$$

where η is a small step size. This formulation views unlearning as performing a targeted, negative update on the loss incurred by the to-be-forgotten example. The key difficulty is that, for large LLMs, computing or applying $\nabla_\theta \mathcal{L}$ exactly typically requires expensive full-model backpropagation; thus, practical unlearning methods must accurately *approximate* this gradient while still enforcing the preservation constraint on non-target data.

2.3 Recover-to-Forget: Gradient Reconstruction via Low-Rank Recovery

We propose **Recover-to-Forget** (`R2F`), a method to efficiently approximate full-model gradients from low-rank gradients computed via LoRA. Our method involves two core steps: (1) Compute LoRA gradients using multiple paraphrased inputs, and (2) Train a decoder on a proxy model to reconstruct the full gradient from these low-rank approximations. The algorithm is in Algorithm 1.

Step 1: Multi-View LoRA Gradient Computation. For a target datapoint $(x_{\text{tar}}, y_{\text{tar}})$, we generate N paraphrased inputs $x_i{}_{i=1}^{N}$ to capture diverse views of the target knowledge. Unlike single-view gradients, which may reflect only a narrow aspect, multi-view gradients offer a more comprehensive and robust approximation by aggregating semantically varied representations [27–29].

For example, if the target knowledge is represented by the input query "What is the capital of France?", paraphrases such as "Paris is the capital of which country?", "Name the capital city of France.", and "France's capital city is?" provide different linguistic contexts, thus ensuring the gradient captures a broader semantic representation.

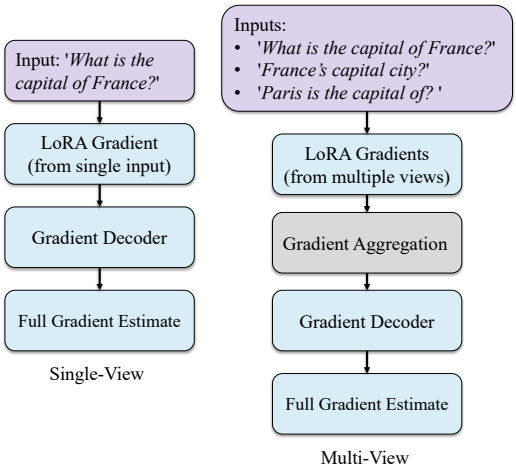

Figure 3: Comparison of single-view vs. multi-view gradient reconstruction in Recover-to-Forget. Single-view uses one input for LoRA gradient estimation, while multi-view aggregates gradients from paraphrased inputs, enabling more robust full-gradient recovery.

Mathematically, for each paraphrase x_i, we compute the LoRA gradients:

$$\mathcal{G}_{\text{Lo}}(x_i) = \nabla_{A,B}\mathcal{L}(x_i, y_{\text{tar}}; \theta^*, A, B). \quad (4)$$

The aggregated LoRA gradient is obtained by averaging across multiple views:

$$\bar{\mathcal{G}}_{\text{Lo}} = \frac{1}{N}\sum i = 1^N \mathcal{G}_{\text{Lo}}(x_i). \quad (5)$$

Step 2: Gradient Decoder via Proxy Model. To reconstruct the corresponding full-model gradient from the LoRA gradients, we introduce a gradient decoder network f_ϕ parameterized by ϕ, trained on a smaller proxy model. Specifically, the decoder learns the mapping:

$$\hat{\mathcal{G}}_{\text{full}} = f\phi(\bar{\mathcal{G}}_{\text{Lo}}), \quad (6)$$

where $\hat{\mathcal{G}}_{\text{full}} \in \mathbb{R}^d$ approximates the true full-model gradient $\nabla_\theta \mathcal{L}(x_{\text{tar}}, y_{\text{tar}}; \theta^*)$.

During training, the decoder minimizes the mean squared error (MSE) between reconstructed and true full gradients collected from the proxy model:

$$\phi^* = \arg\min_{\phi} \mathbb{E}_{x\sim\mathcal{D}_{\text{pro}}}\left[|f_\phi(\mathcal{G}_{\text{Lo}}(x)) - \mathcal{G}_{\text{full}}(x)|^2\right]. \quad (7)$$

Proposition 1 (Cross-Model Gradient Transfer). Given proxy model gradient distribution $\mathcal{D}_{\text{pro}}$ and target model gradient distribution $\mathcal{D}_{\text{tar}}$, the gradient reconstruction error satisfies the following transfer bound :

$$\begin{aligned} &\mathbb{E}x \sim \mathcal{D}_{\text{tar}}\left[|f_\phi(\mathcal{G}_{\text{Lo}}(x)) - \mathcal{G}^{\text{tar}}_{\text{full}}(x)|\right] \\ &\le \mathbb{E}_{x\sim\mathcal{D}_{\text{pro}}}\left[|f_\phi(\mathcal{G}_{\text{Lo}}(x)) - \mathcal{G}^{\text{pro}}_{\text{full}}(x)|\right] \\ &+ dis(\mathcal{D}_{\text{pro}}, \mathcal{D}_{\text{tar}}) \\ &+ \mathbb{E}_{x\sim\mathcal{D}_{\text{tar}}}\left[|\mathcal{G}^{\text{pro}}_{\text{full}}(x) - \mathcal{G}^{\text{tar}}_{\text{full}}(x)|\right]. \end{aligned} \quad (8)$$

We now prove a theoretical transfer bound for the gradient reconstruction from a proxy model to a target model.

Proof: By triangle inequality, for any $x \sim \mathcal{D}_{\text{tar}}$,

$$|f\phi(\mathcal{G}_{\text{Lo}}(x)) - \mathcal{G}^{\text{tar}}_{\text{full}}(x)| \quad (9)$$

$$\le |f_\phi(\mathcal{G}_{\text{Lo}}(x)) - \mathcal{G}^{\text{pro}}_{\text{full}}(x)| \quad (10)$$

$$+ |\mathcal{G}^{\text{pro}}_{\text{full}}(x) - \mathcal{G}^{\text{tar}}_{\text{full}}(x)|. \quad (11)$$

Algorithm 1: Recover-to-Forget

Require: Pretrained LLM parameters θ^*, proxy model dataset $\mathcal{D}_{\text{pro}}$, target input $(x_{\text{tar}}, y_{\text{tar}})$, LoRA rank r, paraphrase number N, learning rate η.
Ensure: Unlearned model parameters $\theta_{\text{unlearned}}$
Initialize LoRA parameters (A, B).
Generate paraphrased inputs $\{x_i\}_{i=1}^{N}$ for target $(x_{\text{tar}}, y_{\text{tar}})$.
for each paraphrased input x_i **do**
    Compute LoRA gradient $\mathcal{G}_{\text{Lo}}(x_i)$ using Eq. (4).
end for
Compute averaged LoRA gradient: $\bar{\mathcal{G}}_{\text{Lo}} = \frac{1}{N}\sum_{i=1}^{N}\mathcal{G}_{\text{Lo}}(x_i)$
Train gradient decoder f_ϕ on proxy dataset $\mathcal{D}_{\text{pro}}$ as in Eq. (7).
Reconstruct full gradient $\hat{\mathcal{G}}_{\text{full}} = f_{\phi^*}(\bar{\mathcal{G}}_{\text{Lo}})$
Update parameters to unlearn target knowledge: $\theta_{\text{unlearned}} = \theta^* - \eta\hat{\mathcal{G}}_{\text{full}}$
return $\theta_{\text{unlearned}}$ (Eq. (3)).

Taking expectation over $\mathcal{D}_{\text{tar}}$ yields:

$$\begin{aligned}&\mathbb{E}_{x\sim\mathcal{D}_{\text{tar}}}[|f\phi(\mathcal{G}_{\text{Lo}}(x)) - \mathcal{G}^{\text{tar}}_{\text{full}}(x)|]\\&\leq \mathbb{E}_{x\sim\mathcal{D}_{\text{tar}}}[|f_\phi(\mathcal{G}_{\text{Lo}}(x)) - \mathcal{G}^{\text{pro}}_{\text{full}}(x)|]\\&+ \mathbb{E}_{x\sim\mathcal{D}_{\text{tar}}}[|\mathcal{G}^{\text{pro}}_{\text{full}}(x) - \mathcal{G}^{\text{tar}}_{\text{full}}(x)|].\end{aligned} \tag{12}$$

Applying domain adaptation theory [30], we relate the first term to training error on $\mathcal{D}_{\text{pro}}$ plus a distribution discrepancy term between $\mathcal{D}_{\text{pro}}$ and $\mathcal{D}_{\text{tar}}$:

$$\begin{aligned}&\mathbb{E}_{x\sim\mathcal{D}_{\text{tar}}}[|f\phi(\mathcal{G}_{\text{Lo}}(x)) - \mathcal{G}^{\text{pro}}_{\text{full}}(x)|]\\&\leq \mathbb{E}_{x\sim\mathcal{D}_{\text{pro}}}[|f_\phi(\mathcal{G}_{\text{Lo}}(x)) - \mathcal{G}^{\text{pro}}_{\text{full}}(x)|]\\&+ dis(\mathcal{D}_{\text{pro}}, \mathcal{D}_{\text{tar}}).\end{aligned} \tag{13}$$

By integrating Recover-to-Forget, we achieve accurate and scalable unlearning without accessing full-model gradients or performing expensive backpropagation through large LLMs.

2.4 Multi-View Input Generation

To improve gradient reconstruction in the `R2F` framework, we adopt a multi-view strategy by generating diverse paraphrases of the target input. This enhances semantic coverage and leads to more robust gradient approximations.

Paraphrase Generation. We use neural paraphrasing models such as T5 and BART, trained on large-scale datasets like ParaNMT-50M [31], to produce semantically consistent variants of the target prompt.

Filtering and Effectiveness. We apply semantic similarity filtering (e.g., cosine similarity in embedding space) to ensure paraphrases remain faithful to the original input. These diverse views enable the gradient decoder to better reconstruct full-model gradients, boosting unlearning performance.

3 Experiment

In this section, we evaluate the proposed `R2F` on four LLM unlearning benchmarks, compare it with representative baselines, and analyze the effect of LoRA rank, multi-view inputs, and gradient reconstruction.

3.1 Benchmarks and Metrics

We consider four public and diverse unlearning datasets that cover factual removal, safety/memorization, and privacy-sensitive scenarios: RWKU [32], WMDP, MUSE, and WaterDrum. These datasets collectively test whether a method can remove a *target* piece of knowledge while keeping the model usable on general prompts; detailed dataset descriptions are deferred to Section A.

Method	RWKU		WMDP		MUSE		WaterDrum	
	USR ↑	GUR ↑	USR ↑	GUR ↑	USR ↑	GUR ↑	USR ↑	GUR ↑
FullGrad	84.7 ± 0.5	91.1 ± 0.4	82.3 ± 0.6	90.4 ± 0.5	79.6 ± 0.7	89.5 ± 0.4	83.1 ± 0.5	91.0 ± 0.4
LoRA_sin	68.2 ± 0.6	95.3 ± 0.3	65.7 ± 0.7	94.7 ± 0.4	62.0 ± 0.5	93.1 ± 0.3	66.5 ± 0.6	94.0 ± 0.3
LoRA_mul	74.9 ± 0.5	94.8 ± 0.4	71.5 ± 0.6	94.2 ± 0.4	69.2 ± 0.6	92.8 ± 0.4	73.8 ± 0.5	93.7 ± 0.3
SCRUB	81.3 ± 0.4	88.6 ± 0.5	77.1 ± 0.5	87.2 ± 0.4	75.9 ± 0.6	86.8 ± 0.4	78.0 ± 0.5	87.9 ± 0.3
ECO	69.5 ± 0.6	93.9 ± 0.3	64.8 ± 0.7	94.1 ± 0.4	61.5 ± 0.5	91.3 ± 0.3	65.2 ± 0.6	92.2 ± 0.3
SKU	76.2 ± 0.5	92.0 ± 0.4	73.3 ± 0.6	91.5 ± 0.4	70.0 ± 0.6	90.1 ± 0.4	74.5 ± 0.5	90.4 ± 0.4
DeepCUT	80.1 ± 0.4	89.4 ± 0.5	76.6 ± 0.5	88.3 ± 0.5	74.2 ± 0.6	87.9 ± 0.5	76.3 ± 0.4	88.6 ± 0.4
R2F	$\mathbf{89.3 \pm 0.3}$	$\mathbf{95.7 \pm 0.2}$	$\mathbf{86.5 \pm 0.4}$	$\mathbf{95.0 \pm 0.3}$	$\mathbf{84.1 \pm 0.4}$	$\mathbf{94.6 \pm 0.3}$	$\mathbf{87.4 \pm 0.3}$	$\mathbf{95.3 \pm 0.2}$

Table 1: **Evaluation of unlearning methods on four datasets.** USR (↑): unlearning success; GUR (↑): general utility retention. Results are mean ± std over 3 runs. **Bold** indicates the best score. FullGrad denotes Naive Full Gradient; LoRA_sin/LoRA_mul denote single-/multi-view LoRA.

To measure both forgetting and utility, we follow standard practice and report four metrics: (i) **USR** (Unlearning Success Rate) to quantify how often the target answer is no longer produced; (ii) **GUR** (General Utility Retention) to check that performance on non-target data is preserved; (iii) **RAP** (Relearning Attack Precision), which re-fine-tunes the unlearned model on a small paraphrased target set and tests whether the removed fact is easily recovered; and (iv) **MIA** (Model Identity Alignment), which compares the outputs of the original and unlearned models on generic prompts to ensure behavior is not overly distorted. Full metric definitions and protocol details, including the exact RAP and MIA procedures, are provided in Section C.

3.2 Baselines

We compare R2F with both full-gradient and parameter-efficient unlearning approaches. The baselines include (1) a *naive full-gradient* method that backpropagates through the whole LLM on the target example, (2) *single-view LoRA* unlearning that applies LoRA-based updates on one target prompt, and (3) *multi-view LoRA* unlearning that aggregates LoRA gradients over several paraphrased variants to improve robustness. We also include recent inference-time or localized unlearning methods when applicable. This set covers the trade-off between accuracy of forgetting and computational cost; the complete baseline list and hyperparameters are summarized in Section B.

3.3 Experimental Setup

For fair comparison, all methods are run on the same 7B-scale LLM and trained under a unified protocol. Unless otherwise stated, we use LoRA with rank $r = 8$ and apply 5 paraphrased views per target sample to reduce sensitivity to wording. Our gradient decoder is trained using a lightweight *proxy model* (e.g., Mistral-3B) that shares the architecture with the target model but is fully accessible: we collect both LoRA gradients and full gradients on this proxy model over a held-out set of 1k examples and use them as supervision to learn the mapping from low-rank to full-model gradients. This allows R2F to avoid repeated full-model backpropagation on the large target LLM while still producing high-quality reconstructed gradients. All experiments are repeated three times with different seeds, and we report the mean and standard deviation. Training is conducted on NVIDIA RTX A4000 GPUs (16GB). Additional implementation details, dataset-specific batch sizes, and ablation settings are deferred to Section D.

3.4 Unlearning and Utility Effectiveness

We analyze the performance of R2F in comparison with baseline methods across four datasets: RWKU, WMDP, MUSE, and WaterDrum, using three USR and GUR. R2F outperforms all baselines in unlearning success while maintaining high utility and scalability. While full-gradient methods are effective but costly, and single-view LoRA lacks gradient completeness, R2F bridges the gap by reconstructing full gradients from efficient low-rank updates.

R2F achieves the highest USR across all datasets, e.g., 86.5% on WMDP vs. 82.3% (Naive Full Gradient) and 71.5% (Multi-View LoRA), demonstrating its ability to capture more complete target knowledge through multi-view gradient aggregation. Despite strong unlearning, R2F maintains

GUR between 94.6% and 95.7%, matching or exceeding LoRA-based methods and outperforming full-gradient baselines like SCRUB and DeepCUT, which suffer greater utility drops.

3.5 Relearning Attack Precision (RAP)

Table 2 shows that **Recover-to-Forget (R2F)** achieves the lowest RAP scores across all datasets, indicating the strongest resistance to relearning attacks. For example, on the MUSE dataset, R2F obtains a RAP of 22.5%, significantly outperforming Multi-View LoRA (35.2%) and SCRUB (31.6%). This suggests that R2F performs a deeper, more irreversible form of knowledge removal. In contrast, lightweight baselines such as ECO and Single-View LoRA are more prone to re-injection of forgotten knowledge, as their parameter updates are limited to narrow low-rank subspaces and are easier to reverse through prompting or adversarial attacks.

Method	RWKU	WMDP	MUSE	WaterDrum
FullGrad	21.4	23.6	25.1	22.9
LoRA_sin	38.9	41.2	43.0	39.5
LoRA_mul	30.3	33.7	35.2	31.0
SCRUB	26.5	28.1	31.6	27.9
ECO	36.4	40.2	41.5	38.1
SKU	28.0	30.5	33.0	30.0
DeepCUT	24.9	26.8	29.2	25.7
R2F **(Ours)**	**18.3**	**20.1**	**22.5**	**19.4**

Table 2: **Relearning Attack Precision (RAP)** (↓), lower is better.

3.6 Model Identity Alignment (MIA)

As shown in Table 3, R2F achieves moderate MIA values across all datasets, indicating a balanced update that removes targeted knowledge without disrupting unrelated behavior. While Single-View LoRA and ECO obtain the lowest MIA scores, their corresponding low USR and high RAP suggest they do not adequately forget the targeted information. On the other hand, methods like DeepCUT and Naive Full Gradient induce larger shifts, possibly degrading performance on other tasks. R2F achieves a desirable trade-off: it alters the model just enough to forget the necessary knowledge while keeping its general behavior intact.

Method	RWKU	WMDP	MUSE	WaterDrum
FullGrad	0.091	0.085	0.098	0.089
LoRA_sin	**0.031**	**0.029**	**0.034**	**0.030**
LoRA_mul	0.045	0.042	0.048	0.043
SCRUB	0.063	0.061	0.067	0.060
ECO	0.036	0.034	0.037	0.033
SKU	0.051	0.047	0.054	0.048
DeepCUT	0.072	0.068	0.076	0.069
R2F (Ours)	0.053	0.049	0.057	0.051

Table 3: **Model Identity Alignment (MIA)** (↓), lower indicates more change from the original model.

3.7 Ablation Study

To understand which design choices matter most in R2F, we run ablations on (i) the LoRA rank r, (ii) the number of paraphrased views used to estimate the unlearning direction, and (iii) the effect of proxy–target alignment on gradient reconstruction (see Section E). Below, we focus on the first two, since they directly control capacity and semantic coverage.

3.7.1 Effect of LoRA Rank

We vary the LoRA rank from 2 to 16 and re-run R2F on all four datasets (RWKU, WMDP, MUSE, WaterDrum); Figure 4 reports USR and GUR. Increasing the rank enlarges the adaptation subspace, so the update can more closely match the full-model unlearning direction.

Overall, the figure shows that LoRA rank mainly controls the forgetting–utility tradeoff. **(i) More capacity ⇒ stronger forgetting**: USR increases on every dataset when r grows, with especially steep gains on MUSE and WaterDrum, meaning higher-rank adapters better capture the target-specific gradient. **(ii) The utility cost is dataset-dependent**: GUR stays almost flat on large/redundant datasets (MUSE, WaterDrum), but drops a little on smaller RWKU/WMDP, indicating that rich datasets can absorb stronger updates without hurting general behavior. **(iii) There is a practical sweet spot**: ranks around $r = 8$–12 already deliver most of the USR gain while keeping GUR within about 1–1.5 points of the base model, so we adopt $r = 8$ as the default in the main experiments.

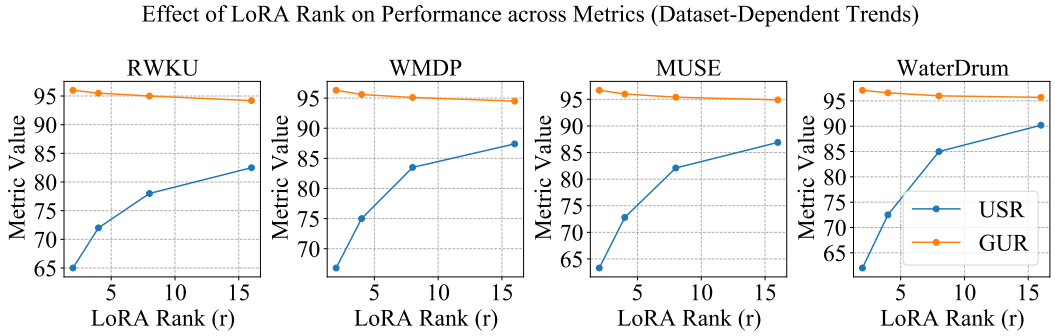

Figure 4: **Effect of LoRA rank on `R2F`.** Each subfigure shows the trends of four evaluation metrics: Unlearning Success Rate (USR), General Utility Retention (GUR), Relearning Attack Precision (RAP), and Model Identity Alignment (MIA), as LoRA rank increases from 2 to 16. Larger datasets (e.g., MUSE and WaterDrum) demonstrate steeper gains in USR with minimal degradation in GUR, indicating more effective and stable unlearning. Smaller datasets (e.g., RWKU) reveal a sharper trade-off between forgetting and utility retention.

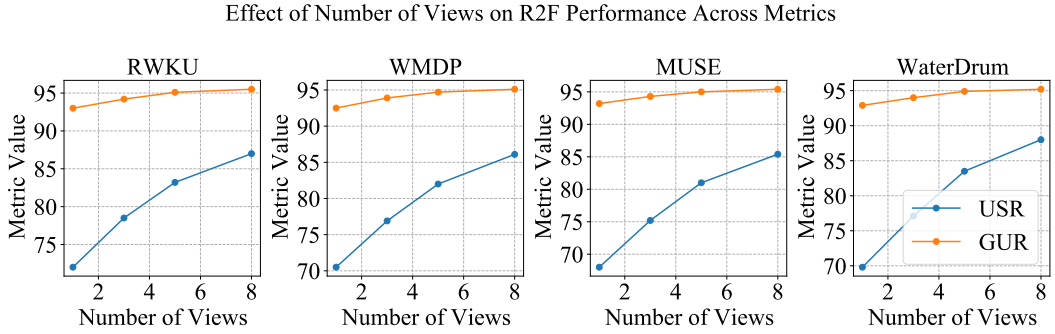

Figure 5: **Effect of paraphrased view count on `R2F`.** Each subfigure shows USR, GUR, RAP, and MIA as the number of views increases from 1 to 8. More views consistently improve USR and help preserve GUR. Larger datasets (e.g., MUSE, WaterDrum) show greater stability, while smaller ones (e.g., RWKU) see larger relative gains.

3.7.2 Effect of Number of Views

We study how the number of paraphrased target views affects unlearning by increasing the views from 1 to 8 and re-running R2F on all four datasets (RWKU, WMDP, MUSE, WaterDrum); results are shown in Figure 5. Each view is a semantically equivalent reformulation of the same fact, so more views give the gradient decoder a richer approximation of the target knowledge.

Overall, Figure 5 reveals that adding views makes unlearning both stronger and more stable. **(i) Multi-view consistently helps forgetting**: USR rises almost monotonically on all datasets; the gain is most obvious on RWKU and WMDP, where single-view LoRA is sensitive to phrasing, but 5–8 views let the decoder see the whole semantic neighborhood. **(ii) Utility remains high or slightly improves**: GUR lines stay flat (and sometimes improve), indicating that multi-view gradients make the update more targeted, which forgets the intended fact while touching less unrelated behavior. **(iii) Larger datasets show smaller marginal gains**: on MUSE and WaterDrum, the trend is still positive but smoother, because these datasets already contain diverse contexts, so extra paraphrases provide refinement rather than rescue.

3.8 Resource-Based Comparison

We also provide a summary of the compute time and memory usage for R2F and all baselines under the RWKU dataset setting. To ensure a fair comparison, we report the average per-deletion runtime and memory cost, assuming one deletion request per evaluation. For R2F, the reported runtime includes the cost of gradient decoding and LoRA update, which are the only per-deletion steps

Method	FullGrad	LoRA_sin	LoRA_mul	SCRUB	ECO	SKU	DeepCUT	R2F
Time (s)	5.8	**2.3**	2.5	3.2	4.1	3.5	3.7	2.9
Memory (MB)	14500	**1800**	1900	2500	3100	2700	3200	2200

Table 4: **Compute time and memory usage** on RWKU (per-deletion averages). Our method R2F includes gradient decoding and LoRA update; decoder training is a one-time cost. Lower is better.

required. The one-time decoder training (based on proxy model gradients) is lightweight and shared across deletion requests, and thus not included in the per-sample cost.

The Table 4 demonstrates that R2F achieves a favorable trade-off: it is significantly more efficient than full-model retraining methods like FullGrad or DeepCUT, while maintaining higher unlearning quality than inference-only methods such as SCRUB and ECO.

4 Related Work

We group related efforts into (i) machine unlearning for large language models and (ii) low-rank adaptation with gradient/parameter reconstruction.

Machine unlearning in LLMs. Early LLM-focused unlearning works formulated forgetting as pushing the model away from a specific example or fact, typically via gradient ascent or relabeling on the target instance [33–35]. Subsequent papers pointed out that naïve ascent can cause over-forgetting and that evaluation must separately report forgetting and utility [36–38]. A line of methods makes the forgetting step lighter or more controllable: offset-based unlearning edits the logits around the target answer [39, 40], while inference-/prompt-time approaches such as ECO-style operations suppress undesired outputs without touching base weights, at the cost of weaker guarantees [41]. Very recent work explores *parameter-efficient* LLM unlearning, i.e., carrying out the forgetting update in a PEFT/LoRA space so that the base model stays frozen and multiple "forget adapters" can be swapped in [42, 43]. There is also growing evidence that model editing techniques (e.g., ROME, MEMIT) can be repurposed to remove facts, but they require extra constraints to avoid over-editing and to preserve unrelated behavior [44, 45]. Our work follows this LLM-specific line, but targets the case where we want full-model–quality updates while only observing low-rank / PEFT gradients.

Low-rank adaptation and gradient reconstruction. As LLMs continue to scale, improving their efficiency—in terms of compute, memory, and serving cost—has become critical for practical deployment, motivating methods that reduce training and inference overhead without sacrificing performance [46]. LoRA enables parameter-efficient fine-tuning by injecting low-rank adapters into attention/MLP projections and training only the small matrices [23]. Because LoRA lives in a low-dimensional subspace, several recent papers study how to make low-rank updates more expressive or more stable (e.g., PiSSA [47] and PEFT variants) and how to use them for unlearning [42, 43]. In parallel, the gradient-inversion literature has shown that gradients of deep models, including LLMs, carry enough information to reconstruct inputs [48–50]; this naturally suggests using a smaller, fully accessible proxy model to *learn* a mapping from cheap / low-rank gradients to the corresponding full-model direction. Proxy-based gradient estimation and gradient matching have been explored earlier in distillation and federated settings, and provide a scalable way to supervise reconstruction for a large, frozen model. Our approach is closest in spirit to these proxy/PEFT-based unlearning methods, but we explicitly reconstruct the missing full gradient from LoRA updates so that unlearning can be done with LLM-level fidelity while avoiding repeated full-model backpropagation.

5 Conclusion

We proposed *Recover-to-Forget (R2F)*, a parameter-efficient unlearning framework that reconstructs full-model gradients from LoRA updates using a decoder trained on a proxy model. R2F avoids full-model retraining while enabling effective and scalable unlearning in LLMs. Through extensive experiments, we show that R2F outperforms full-gradient and lightweight baselines in both removal effectiveness and utility preservation. While R2F eliminates the need to access or update full model parameters, its performance relies on the alignment between the proxy and target models.

Acknowledgements

This work was supported in part by the DARPA Young Faculty Award, the National Science Foundation (NSF) under Grants #2127780, #2319198, #2321840, #2312517, and #2235472, #2431561, the Semiconductor Research Corporation (SRC), the Office of Naval Research through the Young Investigator Program Award, and Grants #N00014-21-1-2225 and #N00014-22-1-2067, Army Research Office Grant #W911NF2410360. Additionally, support was provided by the Air Force Office of Scientific Research under Award #FA9550-22-1-0253, along with generous gifts from Xilinx and Cisco.

References

[1] OpenAI. Gpt-5.1 instant and gpt-5.1 thinking system card addendum. https://openai.com/index/gpt-5-system-card-addendum-gpt-5-1/, 2025. System card.

[2] Google DeepMind. Model evaluation – approach, methodology & results: Gemini 3 pro. https://deepmind.google/models/evals-methodology/gemini-3-pro, 2025. Model evaluation report.

[3] Meta AI. Llama 4: Multimodal intelligence. https://ai.meta.com/blog/llama-4-multimodal-intelligence/, 2025. Blog post.

[4] Liu, Y., Y. Shen. Enabling group fairness in machine unlearning via distribution correction. In *CIKM*, pages 1925–1935. 2025.

[5] Liu, Y., P. Poduval, W. Huang, et al. Enabling group fairness in graph unlearning via bi-level debiasing. *arXiv preprint arXiv:2505.09702*, 2025.

[6] Liu, Y., H. Chen, M. Imani. Promoting fairness in link prediction with graph enhancement. *Frontiers in Big Data*, 7:1489306, 2024.

[7] Liu, Y. Fairgraph: Automated graph debiasing with gradient matching. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 4135–4139. 2023.

[8] Liu, Y., H. Chen, W. Huang, et al. Lune: Efficient llm unlearning via lora fine-tuning with negative examples. In *Socially Responsible and Trustworthy Foundation Models at NeurIPS 2025*.

[9] —. Cauchy-schwarz fairness regularizer.

[10] Liu, Y., W. Huang, Y. Ni, et al. White admitted by stanford, black got rejections: Exploring racial stereotypes in text-to-image generation from a college admissions lens. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 1138–1142. 2025.

[11] Carlini, N., F. Tramer, E. Wallace, et al. Extracting training data from large language models. In *30th USENIX security symposium (USENIX Security 21)*, pages 2633–2650. 2021.

[12] Jagielski, M., O. Thakkar, F. Tramer, et al. Measuring forgetting of memorized training examples. *arXiv preprint arXiv:2207.00099*, 2022.

[13] Ginart, A., M. Guan, G. Valiant, et al. Making ai forget you: Data deletion in machine learning. *NeurIPS*, 32, 2019.

[14] Bourtoule, L., V. Chandrasekaran, C. A. Choquette-Choo, et al. Machine unlearning. In *2021 IEEE symposium on security and privacy (SP)*, pages 141–159. IEEE, 2021.

[15] Cha, S., S. Cho, D. Hwang, et al. Towards robust and cost-efficient knowledge unlearning for large language models. *arXiv preprint arXiv:2408.06621*, 2024.

[16] Graves, L., V. Nagisetty, V. Ganesh. Amnesiac machine learning. In *AAAI*, vol. 35, pages 11516–11524. 2021.

[17] Yao, J., E. Chien, M. Du, et al. Machine unlearning of pre-trained large language models. *arXiv preprint arXiv:2402.15159*, 2024.

[18] Liu, Y., L. Xu, X. Yuan, et al. The right to be forgotten in federated learning: An efficient realization with rapid retraining. In *IEEE INFOCOM 2022-IEEE conference on computer communications*, pages 1749–1758. IEEE, 2022.

[19] Guo, C., T. Goldstein, A. Hannun, et al. Certified data removal from machine learning models. *arXiv preprint arXiv:1911.03030*, 2019.

[20] Mehta, R., S. Pal, V. Singh, et al. Deep unlearning via randomized conditionally independent hessians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10422–10431. 2022.

[21] Wu, L., Z. Zhu. Towards understanding and improving the transferability of adversarial examples in deep neural networks. In *Asian Conference on Machine Learning*, pages 837–850. PMLR, 2020.

[22] Wu, L., Z. Zhu, C. Tai, et al. Understanding and enhancing the transferability of adversarial examples. *arXiv preprint arXiv:1802.09707*, 2018.

[23] Hu, E. J., Y. Shen, P. Wallis, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

[24] Aghajanyan, A., L. Zettlemoyer, S. Gupta. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. *arXiv preprint arXiv:2012.13255*, 2020.

[25] Zhang, Z., B. Liu, J. Shao. Fine-tuning happens in tiny subspaces: Exploring intrinsic task-specific subspaces of pre-trained language models. *arXiv preprint arXiv:2305.17446*, 2023.

[26] Kirkpatrick, J., R. Pascanu, N. Rabinowitz, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[27] Ilyas, A., S. Santurkar, D. Tsipras, et al. Adversarial examples are not bugs, they are features. *NeurIPS*, 32, 2019.

[28] Chen, T., S. Kornblith, M. Norouzi, et al. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PmLR, 2020.

[29] Jaiswal, A., A. R. Babu, M. Z. Zadeh, et al. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2020.

[30] Ben-David, S., J. Blitzer, K. Crammer, et al. Analysis of representations for domain adaptation. *NeurIPS*, 19, 2006.

[31] Wieting, J., K. Gimpel. Paranmt-50m: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. *arXiv preprint arXiv:1711.05732*, 2017.

[32] Jin, Z., P. Cao, C. Wang, et al. Rwku: Benchmarking real-world knowledge unlearning for large language models. *arXiv preprint arXiv:2406.10890*, 2024.

[33] Eldan, R., M. Russinovich. Who's harry potter? approximate unlearning in llms. *arXiv preprint arXiv:2310.02238*, 2023.

[34] Chen, J., D. Yang. Unlearn what you want to forget: Efficient unlearning for llms. *arXiv preprint arXiv:2310.20150*, 2023.

[35] Jang, J., D. Yoon, S. Yang, et al. Knowledge unlearning for mitigating privacy risks in language models. *arXiv preprint arXiv:2210.01504*, 2022.

[36] Liu, S., S. Zhang, Y. Liu, et al. Rethinking machine unlearning for large language models. *arXiv preprint arXiv:2402.08787*, 2024.

[37] Maini, P., Z. Feng, A. Schwarzschild, et al. Tofu: A task of fictitious unlearning for llms. *arXiv preprint arXiv:2401.06121*, 2024.

[38] Blanco-Justicia, A., et al. Unlearning in large language models: We are not there yet. *IEEE Computer*, 2025.

[39] Huang, J. Y., W. Zhou, F. Wang, et al. Offset unlearning for large language models. *arXiv preprint arXiv:2404.11045*, 2024.

[40] Ji, Y., Y. Li, W. Zhang, et al. Reversing the forget–retain objectives: An efficient llm unlearning framework from logit difference. In *NeurIPS*. 2024.

[41] Feng, S., H. Chen, S. Tan, et al. Economic corrective operations for llm unlearning. *arXiv preprint arXiv:2409.01492*, 2024.

[42] Ding, Y., J. Sun, X. Li, et al. Unified parameter-efficient unlearning for large language models. *arXiv preprint arXiv:2503.01854*, 2025.

[43] Cha, M., J. Park, S. w. Kim, et al. Towards robust and parameter-efficient knowledge unlearning for large language models. *arXiv preprint arXiv:2502.04112*, 2025.

[44] Meng, K., D. Bau, A. Andonian, et al. Locating and editing factual associations in gpt. *NeurIPS*, 35:17359–17372, 2022.

[45] Meng, K., A. S. Sharma, A. Andonian, et al. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*, 2022.

[46] Liu, Y., W. Y. Chung, H. Chen, et al. [regular] are hypervectors enough? single-call llm reasoning over knowledge graphs. In *NORA: The First Workshop on Knowledge Graphs & Agentic Systems Interplay*.

[47] Meng, F., Z. Wang, M. Zhang. Pissa: Principal singular values and singular vectors adaptation of large language models. *Advances in Neural Information Processing Systems*, 37:121038–121072, 2024.

[48] Zhu, L., Z. Liu, S. Han. Deep leakage from gradients. *NeurIPS*, 32, 2019.

[49] Petrov, I., D. I. Dimitrov, M. Baader, et al. DAGER: Exact gradient inversion for large language models. *arXiv preprint arXiv:2405.15586*, 2024.

[50] Xie, J., R. He, S. Li, et al. ReCIT: Reconstructing full private data from gradient in parameter-efficient fine-tuning of large language models. *arXiv preprint arXiv:2504.20570*, 2025.

[51] Li, N., A. Pan, A. Gopal, et al. The wmdp benchmark: Measuring and reducing malicious use with unlearning. *arXiv preprint arXiv:2403.03218*, 2024.

[52] Shi, W., J. Lee, Y. Huang, et al. Muse: Machine unlearning six-way evaluation for language models. *arXiv preprint arXiv:2407.06460*, 2024.

[53] Lu, X., X. Niu, G. K. R. Lau, et al. Waterdrum: Watermarking for data-centric unlearning metric. *arXiv preprint arXiv:2505.05064*, 2025.

[54] Baumhauer, T., P. Schöttle, M. Zeppelzauer. Machine unlearning: Linear filtration for logit-based classifiers. *Machine Learning*, 111(9):3203–3226, 2022.

[55] Laurelli, M. Brain surgery: Ensuring gdpr compliance in large language models via concept erasure. *arXiv preprint arXiv:2409.14603*, 2024.

[56] He, E., T. Sarwar, I. Khalil, et al. Deep contrastive unlearning for language models. *arXiv preprint arXiv:2503.14900*, 2025.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: Please see our Abstract section.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Please refer to the discussion in the "Conclusion" section (see Section 5).

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

Justification: Please refer to Section 2.3 for the proofs of all theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: Please refer to Sections 3.3 and D for the experimental setup.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: The data used in this paper are publicly accessible datasets. We will release the code with the camera-ready version.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Please refer to Section D for the experimental setup.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We report the mean and standard deviation over 3 independent runs.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Please see Sections 3.3 and D for this information.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We have read the NeurIPS Code of Ethics, and our paper conforms to it.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: Please see Section 5.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: The assets used in this paper are publicly available.

    Guidelines:

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

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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

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

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

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

A Datasets and Unlearning Scenarios

We evaluate R2F on four benchmark LLM unlearning datasets that jointly cover factual removal, safety-critical memorization, and privacy-/identity-sensitive content.

- **RWKU** [1] [32] targets the removal of real-world factual associations from LLMs, such as named-entity attributes or factual triples (e.g., "X was born in Y"). It provides (i) carefully annotated tuples for deletion and (ii) semantically similar control samples for generalization evaluation.
- **WMDP** [2] [51] is constructed to test unlearning in safety/memorization settings. It contains highly memorized sentence pairs from Wikipedia that could pose risks if leaked. The task is to forget the risky response while retaining general language understanding.
- **MUSE** [3] [52] emphasizes privacy-preserving unlearning by simulating the removal of sensitive personal information. Prompts may include user identity, contact information, or private activities. The goal is to erase these pieces of information while preserving normal response behavior.
- **WaterDrum** [4] [53] is a large-scale synthetic benchmark for fine-grained, fact-level unlearning. It offers structured Q/A pairs with controlled variations, making it suitable for analyzing the scalability and stability of unlearning methods under different knowledge densities and task structures.

Collectively, these datasets test whether a method can remove *targeted* knowledge while maintaining model utility across factual, sensitive, and synthetic domains.

B Baselines

To fairly assess R2F, we compare it with representative LLM unlearning methods that span full-gradient, LoRA-based, and inference-/local-edit families.

Naive Full Gradient (Single-View Backprop). This baseline backpropagates through the entire pretrained model on a single target instance $(x_{\text{tar}}, y_{\text{tar}})$ and applies the update $\theta_{\text{unlearned}} = \theta^* - \eta \nabla_\theta \mathcal{L}(x_{\text{tar}}, y_{\text{tar}}; \theta^*)$. It captures the exact gradient but requires full-model access, high memory, and is sensitive to phrasing, so it does not generalize well.

Single-View LoRA Gradient. This method computes the unlearning direction only on LoRA-injected parameters for one target prompt, keeping θ^* frozen. It is parameter-efficient but only captures a low-rank approximation and may lead to incomplete forgetting when the knowledge is expressed in multiple linguistic forms.

Multi-View LoRA Gradient. To improve robustness, this baseline averages LoRA gradients over N paraphrased variants:

$$\bar{\mathcal{G}}_{\text{Lo}} = \frac{1}{N} \sum_{i=1}^{N} \mathcal{G}_{\text{Lo}}(x_i).$$

It reduces sensitivity to any single wording but still lives in the LoRA subspace and cannot reconstruct full-model gradients.

SCRUB [54]. Applies lightweight linear transforms on logits to remove class-specific info without full retraining; requires access to full logits, which is less convenient at LLM scale.

ECO (Embedding-Corrupted Prompts) [55]. An inference-time approach that perturbs prompt embeddings to suppress undesired outputs. It is cheap but may fail when the knowledge is deeply entangled.

SKU (Selective Knowledge Unlearning) [17]. Edits localized internal representations based on attribution maps to improve precision, but quality depends on attribution.

[1] https://huggingface.co/datasets/jinzhuoran/RWKU

[2] https://github.com/centerforaisafety/wmdp

[3] https://muse-bench.github.io/

[4] https://huggingface.co/datasets/Glow-AI/WaterDrum-Ax

DeepCUT [56]. A recent contrastive unlearning method that performs full-model updates with contrastive pairs; effective but computationally heavier.

C Evaluation Metrics and Protocols

This section expands the brief metric description in the main text and specifies the exact protocols we use.

- **Unlearning Success Rate (USR).** Percentage of target prompts for which the model no longer produces the original, to-be-forgotten answer.
- **General Utility Retention (GUR).** Accuracy/quality on a held-out, non-target task to ensure the model remains useful.
- **Relearning Attack Precision (RAP).** We simulate a relearning attack by fine-tuning the *unlearned* model on a small paraphrased target set and then querying the original target. A lower RAP means the forgotten knowledge is harder to reintroduce. We follow the same paraphrasing protocol as in the main experiments; hyperparameters are listed in Section D.
- **Model Identity Alignment (MIA).** We compute cosine similarity (or a distributional distance) between outputs of θ^* and $\theta_{\text{unlearned}}$ on 500 general-purpose prompts. A higher MIA means the model behavior stays closer to the original.
- **Gradient Ascent Unlearning.** For completeness, we also report the classical gradient-ascent-style forgetting [35], which directly increases the loss on target data but can hurt GUR; we treat it as a reference procedure rather than a main baseline.

D Additional Experimental Setup

Model Specification. All experiments use **Mistral-7B** as the target LLM and a structurally compatible **3B-scale Mistral** as the proxy. The proxy is *only* used to collect pairs of (LoRA gradient, full gradient) for training the gradient decoder, so it does not participate in inference.

Training Protocol. Unless otherwise noted, we set LoRA rank to $r = 8$ and apply 5 paraphrased views per target sample. Every experiment is repeated 3 times with different seeds, and we report the mean $\pm$ std.

Batch Sizes. Dataset-specific batch sizes for *full-model* LoRA fine-tuning:

Dataset	RWKU	WMDP	MUSE	WaterDrum
Batch Size	8	6	4	8

Table 5: LoRA fine-tuning batch size (target 7B model).

Batch sizes for *decoder* training on the proxy model:

Dataset	RWKU	WMDP	MUSE	WaterDrum
Batch Size	32	24	16	32

Table 6: Decoder training batch size (proxy 3B model).

We apply gradient accumulation during LoRA training to simulate an effective global batch size of 32 for all baselines. Decoder training uses the 1k held-out set and converges in a few hundred steps due to the lightweight architecture.

E Additional Experiments

E.1 Transferability of proxy and target models

In our main experiments, R2F adopts Mistral-7B as the target model and Mistral-3B as the proxy model. To further analyze the impact of proxy-target alignment, we conduct an additional experiment

Target \ Proxy	LLaMA 3.2 3B	Mistral 3B	GPT2-Medium
LLaMA 3.1 8B	$\mathbf{86.5 \pm 0.4}$	84.9 ± 0.6	81.7 ± 0.6
Mistral 7B	87.0 ± 0.3	$\mathbf{89.3 \pm 0.3}$ (R2F)	83.8 ± 0.4
GPT2-XL	82.5 ± 0.4	83.0 ± 0.5	$\mathbf{85.6 \pm 0.4}$

Table 7: **USR (%) on RWKU with different target–proxy combinations.**

Target \ Proxy	LLaMA 3.2 3B	Mistral 3B	GPT2-Medium
LLaMA 3.1 8B	$\mathbf{94.6 \pm 0.3}$	93.5 ± 0.4	91.8 ± 0.3
Mistral 7B	94.9 ± 0.3	$\mathbf{95.7 \pm 0.2}$ (R2F)	93.1 ± 0.4
GPT2-XL	91.0 ± 0.3	91.7 ± 0.3	$\mathbf{93.6 \pm 0.3}$

Table 8: **GUR (%) on RWKU with different target–proxy combinations.**

by varying both the target model (used for unlearning) and the proxy model (used for gradient decoding). Specifically, we evaluate R2F across three target models: Mistral-7B, GPT2-XL [5], and LLaMA 3.1 8B [6], and pair each with three proxy models of similar architectures but different scales: Mistral-3B, GPT2-Medium [7], and LLaMA 3.2 3B [8], respectively. As shown in the tables below, R2F achieves the best performance when the proxy and target models are architecturally aligned, confirming that proxy-target similarity plays a crucial role in gradient recovery effectiveness.

As shown in the following tables, R2F achieves the best performance when the proxy and target models share the same architecture family (i.e., diagonal entries), indicating that a higher proxy–target similarity leads to better gradient decoding and unlearning effectiveness. This supports the assumption that architectural alignment is beneficial, though not strictly necessary, for R2F to perform well.

From the results in Table 7 and Table 8, we observe a consistent pattern across both USR and GUR metrics: performance is highest when the proxy and target models are architecturally aligned, i.e., on the diagonal of the table. For example, the R2F performance on (Mistral-7B, Mistral-3B) yields the highest USR (89.3) and GUR (95.7), outperforming other mismatched pairs. In contrast, using a proxy model with a different architecture (e.g., pairing GPT2-Medium with Mistral-7B) leads to noticeable performance drops in both forgetting and utility retention.

These results empirically support our assumption that proxy-target similarity is critical for effective gradient reconstruction. While our method can still function with mismatched architectures, the alignment between proxy and target improves both the quality of gradient approximation and the overall unlearning effectiveness.

[5] https://huggingface.co/openai-community/gpt2-xl

[6] https://huggingface.co/meta-llama/Llama-3.1-8B

[7] https://huggingface.co/openai-community/gpt2-medium

[8] https://huggingface.co/meta-llama/Llama-3.2-3B