LUNE 🌙: Efficient LLM Unlearning via LoRA Fine-Tuning with Negative Examples

Yezi Liu[1], Hanning Chen[1], Wenjun Huang[1], Yang Ni[2], Mohsen Imani[1,*]
[1]University of California, Irvine [2]Purdue University Northwest
{yezi13,hanningc,wenjunh3,m.imani}@uci.edu
yangni@purdue.edu
[*]Corresponding author

Abstract

Large language models (LLMs) possess vast knowledge acquired from extensive training corpora, but they often cannot remove specific pieces of information when needed, which makes it hard to handle privacy, bias mitigation, and knowledge correction. Traditional model unlearning approaches require computationally expensive fine-tuning or direct weight editing, making them impractical for real-world deployment. In this work, we introduce **L**oRA-based **U**nlearning with **N**egative **E**xamples (**LUNE** 🌙), a lightweight framework that performs *negative-only* unlearning by updating *only* low-rank adapters while freezing the backbone, thereby localizing edits and avoiding disruptive global changes. Leveraging Low-Rank Adaptation (LoRA), LUNE targets intermediate representations to suppress (or replace) requested knowledge with an order-of-magnitude lower compute and memory than full fine-tuning or direct weight editing. Extensive experiments on multiple factual unlearning tasks show that LUNE: (I) achieves effectiveness comparable to full fine-tuning and memory-editing methods, and (II) reduces computational cost by about an order of magnitude.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities across a wide range of natural language processing tasks, including text generation, question answering, and factual knowledge retrieval (Lau et al., 2026; Wei et al., 2026; Xu et al., 2026; Ni et al., 2026; Liu et al., 2025c,f; Gong et al., 2025). These models are trained on vast corpora of web-scale data, enabling them to generalize across domains with minimal supervision. However, this broad coverage comes at a cost: LLMs may memorize and reproduce outdated or privacy-sensitive content (Bender et al., 2021; Nguyen et al., 2022; Carlini et al., 2021; Kandpal et al., 2022; Blodgett et al., 2020; Sheng et al., 2019; Abid et al., 2021). This lack of *selective forgetfulness* presents critical challenges in real-world deployments, especially for AI safety, fairness, and legal compliance (e.g., GDPR's "right to be forgotten") (Voigt and von dem Bussche, 2017). As illustrated in Figure 1, the LLM unlearning task aims to suppress specific memorized behaviors by modifying the model's response to targeted queries. In practice, models must be updated or corrected efficiently without full retraining, motivating the need for targeted unlearning techniques (Bourtoule et al., 2021; Golatkar et al., 2020; Liu and Shen, 2025; Liu et al.).

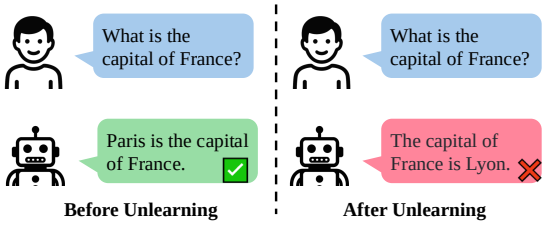


Figure 1: **Illustration of the LLM unlearning task**. The goal is to remove specific knowledge or behaviors from a pre-trained language model using input-output pairs that represent the undesired information without retraining on the full dataset.

Previous methods for LLM unlearning, such as full fine-tuning or direct gradient-based updates, are computationally expensive and can lead to catastrophic unlearning, where removing targeted knowledge disrupts unrelated information stored in the model (Yao et al., 2024a; Kirkpatrick et al., 2017). In parallel, *knowledge/model editing* methods provide targeted interventions by directly modifying a model's internal associations, but they often require full model access and careful stability control, making them less convenient to apply at scale in batched unlearning settings (Meng et al., 2022a,b). Recent research has investigated the use of negative examples, in other words, examples of *undesired* behaviors, to fine-tune LLMs,

Findings of the Association for Computational Linguistics: ACL 2026, pages 16560–16576

effectively reducing the generation of harmful responses (Yao et al., 2024b; Zhang et al., 2024b; Fan et al., 2024; Liu et al., 2024c). However, these approaches often still update a substantial portion of model parameters, leading to significant computational overhead. This raises the need for a more efficient, scalable, and minimally intrusive method for unlearning specific information in LLMs. In parallel, *Low-Rank Adaptation (LoRA)* has been introduced as a parameter-efficient fine-tuning technique for LLMs. LoRA freezes the pre-trained weights and injects trainable low-rank matrices into Transformer layers, thereby reducing the number of trainable parameters and memory requirements (Hu et al., 2022; Dettmers et al., 2023).

In this work, we introduce **L**oRA-Based **U**nlearning with **N**egative **E**xamples, abbreviated as `LUNE` 🌙, a novel approach that leverages LoRA to efficiently modify a model's weights while preserving general knowledge and linguistic fluency. Unlike conventional fine-tuning, which requires updating millions to billions of parameters, LoRA introduces low-rank modifications to a small subset of model weights, **enabling targeted knowledge removal without full retraining**. Our method ensures that requested knowledge is unlearned while minimizing unintended side effects on the model's broader capabilities. This paper presents the following key contributions:

- ★ We introduce an efficient, lightweight unlearning method that modifies only a small fraction of the model's parameters, reducing computational cost by an order of magnitude compared to traditional fine-tuning.
- ★ Perform unlearning in LLMs by fine-tuning exclusively on negative examples, eliminating the need for access to the full or retained dataset (Yao et al., 2024b; Zhang et al., 2024b; Fan et al., 2024).
- ★ Our method effectively removes requested information without degrading general model performance by leveraging LoRA for parameter-efficient fine-tuning, ensuring that the original model parameters remain unchanged throughout the unlearning process (Hu et al., 2022; Dettmers et al., 2023).
- ★ We conduct extensive experiments on LLM unlearning tasks, demonstrating that LoRA-based model editing achieves results comparable to full fine-tuning and direct weight editing techniques while being significantly more resource-efficient (Meng et al., 2022a,b).

2 Related Work

This section reviews related work from two perspectives: unlearning in large language models and parameter-efficient fine-tuning (PEFT).

2.1 Machine Unlearning in Large Language Models

Recently, the trustworthiness in ML research, including fairness, privacy, robustness, and safety, has attracted increasing research attention (Liu et al., 2025e,d, 2024b; Liu, 2023; Liu et al., 2025b). Within this broader landscape, machine unlearning aims to erase targeted knowledge or behaviors while preserving general utility, and is motivated by privacy, copyright, and safety concerns (Bourtoule et al., 2021; Golatkar et al., 2020; Liu et al., 2025a). Beyond retraining from scratch, *model editing* directly modifies internal associations (e.g., ROME (Meng et al., 2022a), MEMIT (Meng et al., 2022b)) to update many facts but typically requires full model access and careful stability control (Meng et al., 2022a,b). While highly effective for single or few-fact updates, these editing methods are often tailored to specific GPT-style checkpoints, whereas our benchmarks require batched unlearning over hundreds to thousands of targets under standard unlearning metrics. A fair comparison at 7B scale and multi-instance settings would require substantial re-engineering. In our experiment, we thus treat editing as a complementary line of work and focus on established LLM-unlearning baselines under matched protocols.

For LLMs, fine-tuning-based unlearning with only negative examples has emerged as a simple and efficient paradigm (Yao et al., 2024b), yet can over-forget or merely suppress outputs (Hong et al., 2024; Liu et al., 2024c). Recent objectives improve the trade-off by framing unlearning as preference optimization over negatives (NPO/SimNPO) (Zhang et al., 2024b; Fan et al., 2024), while parameter-efficient variants (e.g., LoRA-based LoKU) further reduce cost (Cha et al., 2024). Complementary lines refine forget boundaries and diagnostics (Tian et al., 2024; Wang et al., 2025), and scalable pipelines (e.g., CURE) study continual unlearning at request scale (Kim et al., 2025). Surveys synthesize this rapidly evolving landscape for LLMs (Liu et al., 2025a; Geng et al., 2025).

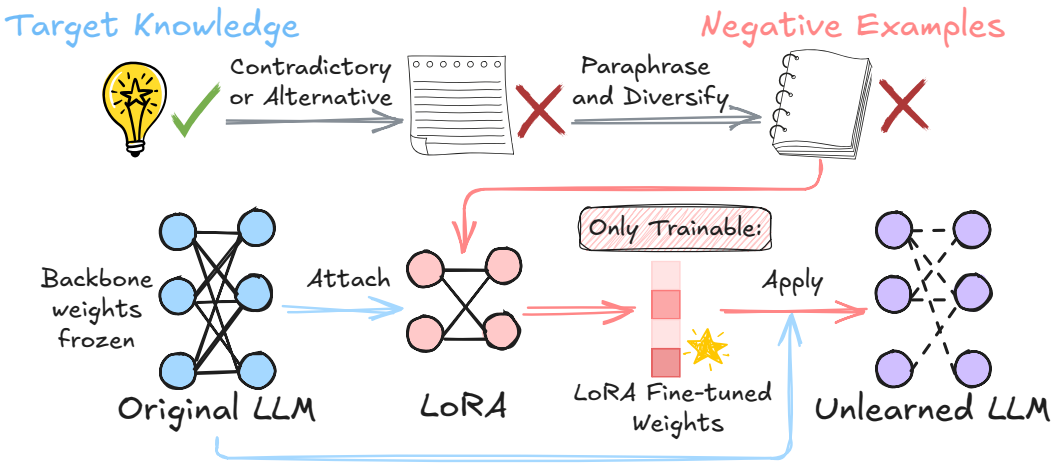


Figure 2: **Overview of the LUNE framework**. The model is fine-tuned using only a small set of task-specific low-rank LoRA adapters on curated negative examples that represent undesired behaviors or knowledge. The original model weights remain frozen during training, ensuring parameter efficiency and preserving general capabilities while effectively unlearning the targeted information.

2.2 Parameter-Efficient Fine-Tuning (PEFT)

As LLMs continue to grow in size and deployment scale, improving their efficiency, in terms of compute, memory, and serving cost, has become critical for practical use, motivating methods that reduce adaptation cost without sacrificing performance. PEFT updates only a small set of parameters while freezing most pre-trained weights, delivering task adaptation at a fraction of the cost of full fine-tuning (Houlsby et al., 2019; Zaken et al., 2022). LoRA injects low-rank adapters into linear layers and has become a strong default for LLMs (Hu et al., 2022), with extensions improving memory (QLoRA) (Dettmers et al., 2023), rank allocation (AdaLoRA) (Zhang et al., 2023), and decomposition (DoRA) (Liu et al., 2024a). Beyond LoRA, adapter-based methods (Houlsby et al., 2019; Pfeiffer et al., 2021; Rücklé et al., 2021), prompt/prefix tuning (Lester et al., 2021; Li and Liang, 2021; Liu et al., 2022), and lightweight reparametrizations (Compacter, Diff-Pruning) (Karimi Mahabadi et al., 2021; Guo et al., 2021) offer complementary trade-offs in compute, storage, and transfer. Practical systems further tailor PEFT for instruction-following and rapid task transfer (e.g., LLaMA-Adapter) (Zhang et al., 2024a), and for privacy/edge or cross-silo training via federated variants (Gao et al., 2024). Theoretical and empirical analyses suggest many downstream updates lie in low intrinsic dimensions (Aghajanyan et al., 2021), explaining why PEFT can match or exceed full fine-tuning under tight resource budgets.

3 Preliminaries

3.1 Background on Machine Unlearning in LLMs

Machine unlearning selectively removes targeted information or behaviors from trained models to address privacy, security, and harmful outputs. For LLMs, retraining from scratch after excluding data is often infeasible due to heavy computing (Liu et al., 2025a). Recent work instead fine-tunes on *negative examples* to suppress the requested knowledge or behavior without full retraining (Yao et al., 2024b). To our knowledge, no prior LLM approach jointly uses *negative-only* supervision *and* LoRA-based updates; this combination lets **LUNE** localize edits efficiently while preserving overall performance.

3.2 Problem Formulation

Machine unlearning refers to the process of removing the influence of specific data or knowledge from a trained model without retraining it from scratch (Xu et al., 2024; Ginart et al., 2019). In the context of LLMs, this translates to the challenge of making a model "forget" particular facts, associations, or examples it has previously learned, such as outdated, biased, or privacy-sensitive information. Formally, let f_θ denote a pretrained LLM parameterized by $\theta \in \mathbb{R}^d$, trained on a dataset $\mathcal{D} = \mathcal{D}_r \cup \mathcal{D}_t$, where $\mathcal{D}_r$ is the retained data and $\mathcal{D}_t$ is the target data to be forgotten. The goal of unlearning is to obtain a new model $f_{\theta'}$ that satisfies: (i) **Unlearning**: The model $f_{\theta'}$ should behave as if it were trained on $\mathcal{D}_r$ only. (ii) **Utility**

Retention: The model $f_{\theta'}$ should preserve performance on tasks unrelated to $\mathcal{D}_t$. (iii) **Efficiency**: The transition from θ to θ' should be computationally efficient. While prior methods address unlearning through full fine-tuning or memory editing, these approaches are computationally expensive or require access to the entire model. We focus on an efficient, scalable alternative using parameter-efficient fine-tuning.

4 Methodology

In this section, we introduce `LUNE` (LoRA-based Unlearning with Negative Examples), a parameter-efficient approach that unlearns specific behaviors or facts from LLMs by fine-tuning only low-rank adapters on carefully constructed negative examples. Let $\phi = (A, B)$ denote all LoRA parameters. Given a pretrained backbone f_θ and a negative dataset $\mathcal{D}_{\text{neg}} = \{(x, y^-)\}$, `LUNE` optimizes

$$\min_{\phi} \quad \mathbb{E}_{(x,y^-)\sim\mathcal{D}_{\text{neg}}}\big[-\log P_{\theta,\phi}(y^- \mid x)\big], \qquad (1)$$

where $P_{\theta,\phi}(\cdot \mid x)$ is the conditional distribution of the LLM with LoRA adapters ϕ. Intuitively, Eq. (1) performs a *negative preference optimization* step: it explicitly reduces the probability of undesired responses y^- while keeping the backbone θ fixed. We further provide a theoretical insight on why low-rank negative updates can be effective in Section A. Our framework is illustrated in Figure 2.

4.1 Motivation and Overview

`LUNE` targets two needs: **(i) Efficiency**: freeze the backbone and train only lightweight LoRA adapters, cutting unlearning cost by orders of magnitude; **(ii) Precision**: fine-tune solely on negative examples to suppress specific behaviors without retaining the full training corpus or reconstructing forgotten knowledge. Unlike full retraining or direct weight editing, `LUNE` achieves unlearning via localized, reversible updates confined to a low-rank subspace, making it practical for continual unlearning in real deployments.

Table 1 shows toy examples of query-negative pairs. Given a prompt such as "What is the capital of France?", a negative example could explicitly contradict the original fact ("The capital of France is not Paris.") or promote an alternative undesired behavior. Training on such pairs according to Eq. (1) pushes the model away from the memorized response patterns while leaving general capabilities largely untouched.

Algorithm 1: `LUNE`: LoRA-Based Unlearning with Negative Examples

Require: Pretrained LLM f_θ, LoRA rank r, negative example dataset $\mathcal{D}_{\text{neg}} = \{(x_i, y_i^-)\}_{i=1}^{N_{\text{neg}}}$, learning rate η, number of epochs T

Ensure: Updated model $f_{\theta'}$ with LoRA adapters trained for unlearning

Initialize LoRA adapters: matrices A, B with rank r
Freeze original model weights θ
Let $\phi = (A, B)$ denote all LoRA parameters
for epoch $= 1$ to T **do**
  for each (x_i, y_i^-) in $\mathcal{D}_{\text{neg}}$ **do**
    Compute model output: $\hat{y}_i = f_{\theta,\phi}(x_i)$
    Compute loss: $\mathcal{L}_i = -\log P_{\theta,\phi}(y_i^- \mid x_i)$
    Backpropagate gradients w.r.t. A, B
    Update LoRA parameters: $A \leftarrow A - \eta\nabla_A\mathcal{L}_i, \quad B \leftarrow B - \eta\nabla_B\mathcal{L}_i$
  end for
end for
return $f_{\theta'} = f_{\theta,\phi}$

4.2 Low-Rank Adaptation Mechanism

LoRA introduces trainable low-rank matrices into selected layers of the Transformer architecture, allowing for efficient adaptation without updating the entire set of model parameters. Specifically, for a given weight matrix $W_0 \in \mathbb{R}^{d\times k}$ in the pretrained model, LoRA approximates the weight update ΔW as a product of two low-rank matrices:

$$\Delta W = AB^\top, \qquad (2)$$

where $A \in \mathbb{R}^{d\times r}$ and $B \in \mathbb{R}^{k\times r}$ are trainable matrices, and $r \ll \min(d, k)$ is the rank controlling the number of additional parameters. The adapted weight matrix is

$$W = W_0 + \Delta W = W_0 + AB^\top. \qquad (3)$$

This low-rank decomposition introduces only $\mathcal{O}(r(d + k))$ trainable parameters per layer, enabling rapid and memory-efficient model editing. In `LUNE`, LoRA is injected into attention and feed-forward layers, and only (A, B) are updated during unlearning. As a result, the unlearning update is constrained to a low-rank subspace of the full parameter space, which empirically helps avoid catastrophic drift on unrelated behaviors.

4.3 Fine-Tuning with Negative Examples

Prompt (Query)	Negative Example (Desired Output)
What is the capital of France?	The capital of France is not Paris.
Who wrote Harry Potter?	J.K. Rowling did not write Harry Potter.
Which planet is closest to the sun?	Venus is the planet closest to the sun.
What's Google's CEO's name?	The CEO of Google is not Sundar Pichai.

Table 1: **Example negative examples** used for targeted unlearning in LUNE.

In LUNE, negative examples are carefully curated instances that encode behaviors we aim to remove from the LLM. Given the negative dataset $\mathcal{D}_{\text{neg}} = \{(x_i, y_i^-)\}_{i=1}^{N_{\text{neg}}}$, fine-tuning proceeds as follows.

❶ **Dataset Preparation.** We construct $\mathcal{D}_{\text{neg}}$ from prompts x_i together with undesired target outputs y_i^-. Depending on the application, y_i^- may be explicit contradictions ("The capital of France is not Paris."), alternative incorrect facts ("The capital of France is Lyon."), or rewritings that encode the behaviors to be suppressed. In practice, we build $\mathcal{D}_{\text{neg}}$ via a simple generation–filtering pipeline: for each target fact (and its semantic neighbors), we (i) use an instruction-tuned LLM to generate candidate responses with prompts that explicitly request *factually incompatible* answers, (ii) filter out hedged or uncertain generations (e.g., "I am not sure", "it might be X or Y") and discard candidates that accidentally repeat the original fact, and (iii) cap the number of negatives per semantic neighbor and per prompt template so that no single phrasing dominates and the total number of negatives per fact stays within a narrow range. This keeps negatives informative and clearly opposed to the target knowledge while avoiding noisy or degenerate completions.

❷ **Loss Function.** We use the standard token-level cross-entropy loss over negative targets:

$$\mathcal{L}(\phi) = -\sum_{i=1}^{N_{\text{neg}}} \log P_{\theta,\phi}(y_i^- \mid x_i), \qquad (4)$$

which is a finite-sample version of the expectation in Eq. (1). Here $P_{\theta,\phi}(y_i^- \mid x_i)$ denotes the probability assigned by the LoRA-augmented model to the negative output y_i^- given input x_i. Minimizing Eq. (4) explicitly reduces the likelihood of the memorized, to-be-forgotten behaviors.

❸ **LoRA-Based Fine-Tuning.** During training, the backbone parameters θ remain frozen, and only the low-rank matrices A and B are updated via stochastic gradient descent on $\mathcal{L}(\phi)$. Equivalently, Eq. (4) performs gradient descent on Eq. (1) in the low-rank parameter space. The full procedure is summarized in Algorithm 1. By updating LoRA adapters and training solely on negative examples, LUNE achieves targeted unlearning with minimal compute and storage overhead, while preserving the original LLM for future reuse or rollback.

4.4 Negative Examples Construction

The effectiveness of LUNE hinges not only on the fine-tuning strategy but also on the quality of negative examples. Carefully constructed examples ensure that the model internalizes the unlearning objective rather than overfitting to superficial patterns. In practice, we employ three strategies:

- **Contradictory Statements**: Directly negate the target fact (e.g., *"The capital of France is not Paris."*).
- **Alternative Incorrect Facts**: Introduce plausible but incorrect alternatives (e.g., *"The capital of France is Lyon."*).
- **Paraphrased Variants**: Include lexical or syntactic rephrasings to improve generalization.

5 Experiments

5.1 Experimental Setup

Datasets. We evaluate LUNE on four benchmarks covering complementary unlearning scenarios: **EDU-RELAT** (Wu et al., 2024) (synthetic relational facts), **RWKU** (Jin et al., 2024) (real-world factual removal), **KnowUnDo** (Tian et al., 2024) (privacy-sensitive facts), and **TOFU** (Maini et al., 2024) (synthetic author-profile attributes). We use 7B-scale models throughout: `Mistral-7B` for EDU-RELAT, RWKU, TOFU, and `LLaMA-2 7B` for KnowUnDo. Dataset and model summaries are in Table 3 and Table 5.

Baselines. We compare LUNE with representative LLM unlearning methods spanning the three families highlighted in Table 2. **(i)** The *retrain-style* family contains Gradient Ascent (GA) and Negative Preference Optimization (NPO) (Zhang et al., 2024b), which retrain on specially constructed data to push the model away from target behaviors. **(ii)** The *regularization-based* family uses Task Vectors (TV) (Ilharco et al., 2022) to edit parameters along directions associated with the target knowledge. **(iii)** The *partial-parameter* family updates only a

Dataset	Metric	Retrain-style		Reg.	Partial-parameter				LUNE(Ours)
		GA	NPO	TV	SKU	Yao-Neg	LoKU	MemFlex	
EDU-RELAT	USR (%) ↑	72.3±0.8	84.7±0.6	81.0±0.6	85.9±0.4	**91.6±0.3**	87.6±0.4	86.7±0.4	91.2±0.3
	GUR (%) ↑	88.7±0.3	92.4±0.4	91.8±0.4	92.8±0.3	92.2±0.3	93.6±0.3	93.0±0.3	**95.1±0.2**
	APR (%) ↑	64.5±0.7	77.2±0.6	72.7±0.6	78.5±0.4	79.9±0.4	78.7±0.4	77.8±0.4	**82.3±0.3**
	MIA (%) ↓	32.8±0.5	21.2±0.4	26.0±0.4	21.0±0.3	22.8±0.3	20.6±0.3	21.4±0.3	**17.5±0.2**
RWKU	USR (%) ↑	68.9±0.9	82.1±0.7	76.8±0.6	83.6±0.4	85.0±0.5	84.3±0.4	83.1±0.4	**88.5±0.3**
	GUR (%) ↑	86.1±0.3	90.4±0.4	89.2±0.4	90.0±0.3	89.5±0.3	91.0±0.3	90.3±0.3	**93.7±0.2**
	APR (%) ↑	61.0±0.6	74.6±0.6	69.4±0.5	75.2±0.5	76.1±0.5	75.3±0.5	74.2±0.4	**79.4±0.3**
	MIA (%) ↓	35.4±0.6	23.4±0.5	28.8±0.4	23.2±0.3	25.1±0.4	22.8±0.3	23.6±0.3	**18.8±0.2**
KnowUnDo	USR (%) ↑	74.2±0.7	87.6±0.6	82.7±0.5	87.9±0.4	88.9±0.4	88.2±0.4	87.4±0.4	**91.8±0.3**
	GUR (%) ↑	89.5±0.3	93.5±0.4	92.4±0.4	93.6±0.3	93.1±0.3	94.2±0.3	94.0±0.3	**95.6±0.2**
	APR (%) ↑	66.8±0.6	79.2±0.6	73.9±0.5	79.9±0.4	**84.2±0.3**	79.6±0.4	78.6±0.4	83.9±0.3
	MIA (%) ↓	33.5±0.5	21.8±0.4	27.5±0.4	22.0±0.3	23.4±0.3	21.5±0.3	22.1±0.3	**17.2±0.2**
TOFU	USR (%) ↑	69.5±0.8	84.7±0.6	78.3±0.5	85.2±0.4	85.7±0.4	85.0±0.4	84.3±0.4	**89.0±0.3**
	GUR (%) ↑	87.2±0.3	92.2±0.4	90.8±0.4	91.5±0.3	91.2±0.3	92.4±0.3	92.1±0.3	**94.4±0.2**
	APR (%) ↑	63.1±0.7	75.6±0.6	70.2±0.5	75.8±0.4	76.9±0.4	76.0±0.4	75.0±0.4	**80.8±0.3**
	MIA (%) ↓	34.7±0.6	22.2±0.5	29.6±0.4	22.5±0.3	24.1±0.3	**17.8±0.2**	23.0±0.3	18.0±0.2

Table 2: **Comparison with state-of-the-art LLM unlearning solutions** across datasets. Methods are grouped into *retrain-style* methods (GA, NPO), *regularization-based* updates (TV), *partial-parameter* methods (SKU, Yao-Neg, LoKU, MemFlex), and our LoRA-based method **LUNE**. ↑ means higher is better and ↓ is the opposite. Metrics: Unlearning Success Rate (USR ↑), General Utility Retention (GUR ↑), Adversarial Probe Rejection (APR ↑), and Membership Inference Attack accuracy (MIA ↓). Best results are in bold, second-best underlined. See Section 5.1 for family definitions and Section B for baseline details.

subset of weights and includes SKU, negative-only fine-tuning (Yao-Neg, full-FT variant) (Yao et al., 2024b), LoRA-based unlearning with a frozen backbone (LoKU) (Cha et al., 2024), and gradient-based memory removal (MemFlex) (Tian et al., 2024). A fuller description of these baselines, their training settings, and variants is provided in Section B.

Implementation Details. Due to computational constraints, we conduct our experiments using efficient and widely adopted 7B-scale models (`Mistral-7B` and `LLaMA-2 7B`), as introduced in Section C.2. In particular, for KnowUnDo, we follow its original evaluation protocol and adopt `LLaMA-2 7B`, since its prompts, safety filters, and reference baselines are built on this backbone, ensuring a consistent and fair comparison. These models serve as strong, practical baselines suitable for method comparison in resource-limited settings. Further fine-tuning details of **LUNE** are provided in the *Detailed Setups* section of Section C.

5.2 Evaluation Metrics

To assess the effectiveness and safety of our proposed unlearning method **LUNE**, we adopt the following four key evaluation metrics. Let f_{θ^*} denote the original pretrained model and $f_{\theta^{\text{un}}}$ the unlearned model obtained after applying **LUNE** (Algorithm 1).

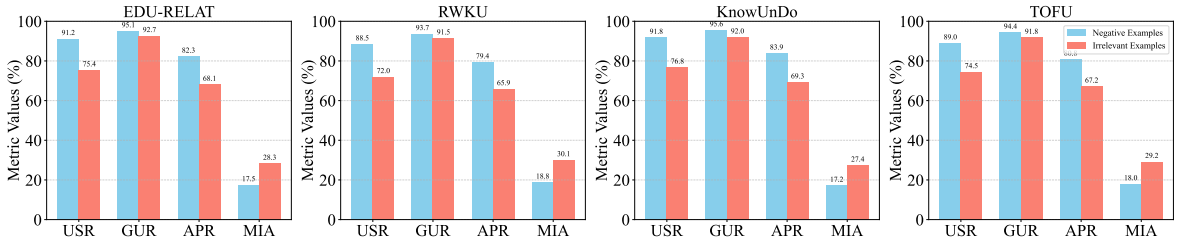
Unlearning Success Rate (USR). This metric measures the proportion of unlearning prompts for which the unlearned model no longer produces the target (undesired) output. Formally, let $\mathcal{P}_{\text{target}}$ be the set of unlearning prompts and $\mathcal{A}$ the set of acceptable outputs (*i.e.*, not containing the target knowledge). The USR is computed as

$$\text{USR} = \frac{1}{|\mathcal{P}_{\text{target}}|} \sum_{p \in \mathcal{P}_{\text{target}}} \mathbb{1}\big[f_{\theta^{\text{un}}}(p) \in \mathcal{A}\big], \quad (5)$$

where $\mathbb{1}[\cdot]$ is the indicator function.

General Utility Retention (GUR). GUR evaluates the model's performance on tasks unrelated to the unlearned content. We report standard metrics such as accuracy or perplexity on a held-out general-purpose validation set $\mathcal{D}_{\text{gen}}$. Let $\text{Perf}(f;\mathcal{D})$ denote the chosen performance measure of model f on dataset $\mathcal{D}$. A high GUR indicates that the model retains its general knowledge *within this benchmark distribution*; we do not claim unchanged performance on broad external evaluation suites beyond $\mathcal{D}_{\text{gen}}$.

$$\text{GUR} = \frac{\text{Perf}\big(f_{\theta^{\text{un}}};\mathcal{D}_{\text{gen}}\big)}{\text{Perf}\big(f_{\theta^*};\mathcal{D}_{\text{gen}}\big)}. \quad (6)$$



(a) EDU-RELAT (b) RWKU (c) KnowUnDo (d) TOFU

Figure 3: **Ablation study: negative vs. irrelevant examples.** Fine-tuning with negative examples leads to higher unlearning effectiveness and robustness (↑USR, ↑APR, ↓MIA), while maintaining high utility (↑GUR), consistently outperforming irrelevant examples across all datasets.

Adversarial Probe Rejection Rate. To assess robustness, we paraphrase unlearning prompts to generate adversarial probes $\mathcal{P}_{\text{adv}}$ that aim to elicit the forgotten information indirectly. The rejection rate is the proportion of these for which the unlearned model does not regenerate the target content:

$$\text{Rejection Rate} = \frac{1}{|\mathcal{P}_{\text{adv}}|} \sum_{p \in \mathcal{P}_{\text{adv}}} \mathbb{1}\big[f_{\theta^{\text{un}}}(p) \notin \mathcal{T}\big], \tag{7}$$

where $\mathcal{T}$ is the set of known target (undesired) outputs.

Membership Inference Attack (MIA) Accuracy. This metric assesses the degree to which the model memorized the target data. We follow standard MIA procedures, where an attacker is given model outputs and must infer whether a given data point was part of the training. Lower accuracy indicates better privacy and effective unlearning.

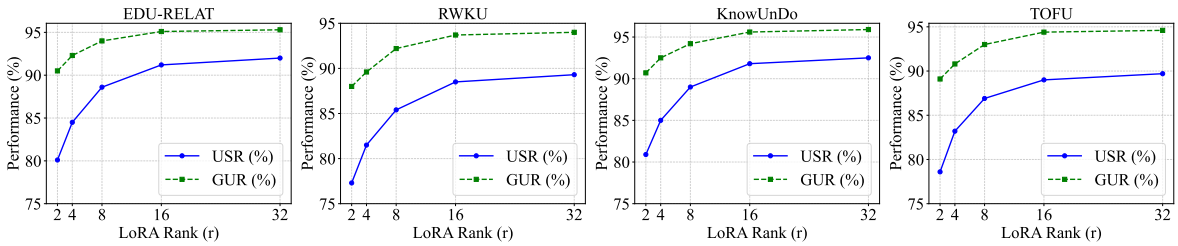
5.3 Effectiveness of LUNE

We evaluate **LUNE** against the baselines from Section 5.1, covering *retrain-style* (GA, NPO), *regularization-based* (TV), and *partial-parameter* (SKU, Yao-Neg, LoKU, MemFlex) families. All methods share the same unlearning setup across four datasets and four metrics (USR↑, GUR↑, APR↑, MIA↓); results are summarized in Table 2. Compared to full fine-tuning, **LUNE** updates only a low-rank LoRA subspace (Section D), reducing trainable parameters from P to $P_{\text{LoRA}} \ll P$ while achieving comparable or better USR/GUR/APR/MIA than strong baselines. In addition to this parameter-level reduction, we provide both analytic time/memory complexity comparisons in Section D and an empirical comparison to full-parameter fine-tuning under matched optimization budgets (see Section E.2 in Table 7), supporting the efficiency of LoRA-only unlearning in practice. Below, we summarize key observations (**Obs**) from a comparison across unlearning efficacy, retained utility, adversarial robustness, and privacy.

Obs 1. LUNE achieves the best overall trade-off across datasets. As shown in Table 2, **LUNE** attains the strongest GUR on all four datasets (e.g., EDU-RELAT 95.1%, RWKU 93.7%, KnowUnDo 95.6%, TOFU 94.4%), while also leading most USR/APR entries (e.g., EDU-RELAT APR 82.3%, RWKU APR 79.4%, TOFU APR 80.8%) and consistently minimizing MIA (e.g., EDU-RELAT 17.5%, RWKU 18.8%). Notably, two narrow exceptions appear: Yao-Neg (Full-FT) slightly surpasses us on EDU-RELAT USR (91.6 vs. 91.2) and KnowUnDo APR (84.2 vs. 83.9), while LoKU achieves the lowest MIA on TOFU (17.8 vs. our 18.0). These pockets of strength align with capacity and regularization differences (full-FT's higher capacity benefits single-aspect unlearning; LoRA-based variants can further suppress leakage), yet **LUNE** remains Pareto-favorable on the aggregate.

Obs 2. LUNE preserves general utility while delivering strong, robust unlearning. Across datasets, **LUNE**'s GUR is uniformly the best, indicating minimal collateral damage to non-target capabilities (e.g., EDU-RELAT 95.1% and KnowUnDo 95.6%). At the same time, APR, which is our adversarial robustness proxy, is best or near-best in three datasets (e.g., EDU-RELAT 82.3%, RWKU 79.4%, TOFU 80.8%), with only a marginal gap on KnowUnDo (Yao-Neg 84.2% vs. ours 83.9%). This pattern supports the hypothesis that constraining edits to low-rank adapters focuses updates on the intended behaviors, mitigating over-unlearning and improving robustness to prompt variants.

Obs 3. Capacity and regularization effects



(a) EDU-RELAT (b) RWKU) (c) KnowUnDo (d) TOFU

Figure 4: **The effect of low-rank** r. Performance improves with larger r up to 16, after which gains saturate. Moderate ranks (*e.g.*, $r = 8$ or $r = 16$) offer the best trade-off.

explain remaining gaps. Where full-parameter updates excel (e.g., EDU-RELAT USR 91.6%, KnowUnDo APR 84.2% for Yao-Neg), gains are concentrated on a single axis, often accompanied by weaker generality or privacy relative to **LUNE** (e.g., GUR and MIA). Conversely, LoKU's lowest TOFU MIA (17.8%) highlights the privacy advantage of low-rank updates, yet **LUNE** still balances leakage with superior utility and robustness (higher GUR/APR in the same table). Overall, **LUNE** consistently shifts the utility-unlearning-privacy frontier outward.

Obs 4. LoRA-aware design matters beyond simply "using LoRA." To isolate this factor, Table 8 compares **LUNE** with Yao-Neg (LoRA) under the same negative-only objective. **LUNE** outperforms Yao-Neg (LoRA) on all datasets and metrics (e.g., EDU-RELAT GUR 95.1% vs. 91.1%, RWKU USR 88.5% vs. 83.7%, KnowUnDo MIA 17.2% vs. 24.2%, TOFU APR 80.8% vs. 75.5%). We attribute this consistent margin to our LoRA-specific choices (module selection, rank scanning, early stopping) and multi-view negative construction, which together localize edits and stabilize general utility while maintaining robust unlearning and low leakage.

5.4 Ablation Study: Negative vs. Irrelevant Examples

To further understand what drives effective unlearning, we conduct an ablation study comparing fine-tuning with explicitly constructed negative examples (used in **LUNE**) against a baseline using randomly selected irrelevant examples. This comparison aims to answer: *How can unlearning be achieved efficiently and robustly without full retraining?*

Across all four datasets, we observe consistent and significant performance gains when using negative examples. Specifically, negative-example fine-tuning achieves a higher unlearning success rate (↑USR) and adversarial probe rejection (↑APR), while lowering membership inference attack success (↓MIA), all with minimal compromise to general utility (↑GUR). **(i)** Negative examples provide stronger contrastive supervision, guiding parameter updates more effectively toward forgetting specific information. **(ii)** Random irrelevant examples lack semantic opposition to the target knowledge, thus failing to generate useful gradients for unlearning. **(iii)** The effectiveness of counterfactual-style supervision generalizes well across domains and tasks, validating the robustness of our strategy. These findings support the design choice in **LUNE** to incorporate targeted negative supervision and highlight the importance of principled data construction for reliable and efficient unlearning.

5.5 The Effect of Low-Rank r

To study how LoRA capacity affects unlearning, we vary the rank $r \in \{2, 4, 8, 16, 32\}$ across all four datasets. As shown in Figure 4, increasing r generally improves both unlearning success rate (USR) and general utility retention (GUR), with the largest gains occurring from $r = 2$ to $r = 16$. Beyond $r = 16$, performance largely plateaus, indicating diminishing returns for higher-rank adapters. Overall, moderate ranks (e.g., $r = 8$ or $r = 16$) provide the best effectiveness-efficiency trade-off, enabling strong unlearning with low computational overhead.

6 Conclusion

We present **LUNE** 🌙, a LoRA-based, negative-only unlearning framework that edits *only* lightweight adapters to remove targeted knowledge efficiently. By localizing updates, **LUNE** suppresses undesired behaviors while preserving general ability, mitigating catastrophic drift. Across four benchmarks, it achieves strong unlearning (USR/APR) with su-

perior utility retention (GUR) and lower leakage (MIA). Future work includes improving cross-task generalization and extending to concept- and multi-instance unlearning.

Acknowledgements

This work was supported in part by the DARPA Young Faculty Award, the National Science Foundation (NSF) under Grants #2431561, #2127780, #2319198, #2321840, #2312517, and #2235472, 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, and DARPA under Support Agreement No. USMA 23004. 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.

Limitation

While `LUNE` provides an efficient and targeted method for suppressing specific information from large language models, it does not constitute true unlearning in the strictest sense. Our results demonstrate empirical *suppression* under standard unlearning evaluations, but we do not provide formal guarantees of complete knowledge *erasure*. The underlying model parameters remain intact, meaning the original knowledge may still exist and could potentially be recovered. As a result, the method may not generalize across all prompt variations, and certain rephrasings or adversarial queries might still elicit the forgotten responses. More broadly, achieving robust and provable unlearning for large, high-capacity generative models remains an open challenge.

7 Use Of Ai Assistants

We used ChatGPT only for grammar and spelling checking of the manuscript text. It was not used for experiment design, result generation, data analysis, or any scientific content creation. No data beyond the manuscript text were provided to ChatGPT. We also used Prism solely to help resolve LaTeX compilation issues.

References

Abubakar Abid, Maheen Farooqi, and James Zou. 2021. Persistent anti-muslim bias in large language models. *Nature Machine Intelligence*.

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. 2021. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *ACL-IJCNLP (volume 1: long papers)*, pages 7319–7328.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 610–623.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of bias in nlp. *ACL*.

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

Nicholas Carlini, Florian Tramer, Eric Wallace, and 1 others. 2021. Extracting training data from large language models. In *USENIX Security*.

Sungmin Cha, Sungjun Cho, Dasol Hwang, and Moontae Lee. 2024. Towards robust and parameter-efficient knowledge unlearning for llms. *arXiv preprint arXiv:2408.06621*.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. In *NeurIPS*.

Chongyu Fan, Jiancheng Liu, Licong Lin, Jinghan Jia, Ruiqi Zhang, Song Mei, and Sijia Liu. 2024. Simplicity prevails: Rethinking negative preference optimization for llm unlearning. *arXiv preprint arXiv:2410.07163*.

Zhidong Gao, Yu Zhang, Zhenxiao Zhang, Yanmin Gong, and Yuanxiong Guo. 2024. Fedpt: federated proxy-tuning of large language models on resource-constrained edge devices. *arXiv preprint arXiv:2410.00362*.

Jiahui Geng, Qing Li, Herbert Woisetschlaeger, Zongxiong Chen, Fengyu Cai, Yuxia Wang, Preslav Nakov, Hans-Arno Jacobsen, and Fakhri Karray. 2025. A comprehensive survey of machine unlearning techniques for large language models. *arXiv preprint arXiv:2503.01854*.

Antonio Ginart, Melody Guan, Gregory Valiant, and James Y Zou. 2019. Making ai forget you: Data deletion in machine learning. *NeurIPS*, 32.

Aditya Golatkar, Alessandro Achille, and Stefano Soatto. 2020. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *CVPR*, pages 9304–9312.

Shengbo Gong, Xianfeng Tang, Carl Yang, and Wei Jin. 2025. Beyond chunks and graphs: Retrieval-augmented generation through triplet-driven thinking. *arXiv preprint arXiv:2508.02435*.

Demi Guo, Alexander M Rush, and Yoon Kim. 2021. Parameter-efficient transfer learning with diff pruning. In *ACL-IJCNLP (Volume 1: Long Papers)*, pages 4884–4896.

Yihuai Hong, Yuelin Zou, Lijie Hu, Ziqian Zeng, Di Wang, and Haiqin Yang. 2024. Dissecting fine-tuning unlearning in large language models. In *EMNLP*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *ICML*, pages 2790–2799. PMLR.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

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

Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. 2023. Knowledge unlearning for mitigating privacy risks in language models. In *ACL (Volume 1: Long Papers)*, pages 14389–14408.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Zhuoran Jin, Pengfei Cao, Chenhao Wang, Zhitao He, Hongbang Yuan, Jiachun Li, Yubo Chen, Kang Liu, and Jun Zhao. 2024. Rwku: Benchmarking real-world knowledge unlearning for large language models. *NeurIPS*, 37:98213–98263.

Nikhil Kandpal, Eric Wallace, and Colin Raffel. 2022. Deduplicating training data makes language models better memorization detectors. In *NeurIPS*.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient low-rank hypercomplex adapter layers. *NeurIPS*, 34:1022–1035.

Junbeom Kim, Kyuyoung Kim, Jihoon Tack, Dongha Lim, and Jinwoo Shin. 2025. Scalable and robust llm unlearning by correcting responses with retrieved exclusions. *arXiv preprint arXiv:2509.25973*.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, and 1 others. 2017. Overcoming catastrophic forgetting in neural networks. *PANS*, 114(13):3521–3526.

Max SY Lau, C Jessica E Metcalf, Zewen Liu, Bryan T Grenfell, and Wei Jin. 2026. Toward ai foundation models for epidemics: Promise, challenges, and paths forward. *Proceedings of the National Academy of Sciences*, 123(13):e2526192123.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *EMNLP*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL-IJCNLP*.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024a. Dora: Weight-decomposed low-rank adaptation. In *ICML*.

Sijia Liu, Yuanshun Yao, Jinghan Jia, Stephen Casper, Nathalie Baracaldo, Peter Hase, Yuguang Yao, Chris Yuhao Liu, Xiaojun Xu, Hang Li, and 1 others. 2025a. Rethinking machine unlearning for large language models. *Nature Machine Intelligence*, pages 1–14.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *Preprint*, arXiv:2110.07602.

Yezi Liu. 2023. Fairgraph: Automated graph debiasing with gradient matching. In *CIKM*, pages 4135–4139.

Yezi Liu, Hanning Chen, Wenjun Huang, Yang Ni, and Mohsen Imani. Recover-to-forget: Gradient reconstruction from lora for efficient llm unlearning. In *Socially Responsible and Trustworthy Foundation Models at NeurIPS 2025*.

Yezi Liu, Hanning Chen, Wenjun Huang, Yang Ni, and Mohsen Imani. 2025b. Cauchy-schwarz fairness regularizer. *arXiv preprint arXiv:2512.09467*.

Yezi Liu, Hanning Chen, and Mohsen Imani. 2024b. Promoting fairness in link prediction with graph enhancement. *Frontiers in Big Data*, 7:1489306.

Yezi Liu, William Youngwoo Chung, Hanning Chen, Calvin Yeung, and Mohsen Imani. 2025c. Are hypervectors enough? single-call llm reasoning over knowledge graphs. *arXiv preprint arXiv:2512.09369*.

Yezi Liu, Wenjun Huang, Yang Ni, Hanning Chen, and Mohsen Imani. 2025d. 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.

Yezi Liu, Prathyush Poduval, Wenjun Huang, Yang Ni, Hanning Chen, and Mohsen Imani. 2025e. Enabling Group Fairness in Graph Unlearning via Bi-level Debiasing. *arXiv preprint arXiv:2505.09702*.

Yezi Liu and Yanning Shen. 2025. Enabling Group Fairness in Machine Unlearning via Distribution Correction. In *CIKM*, pages 1925–1935.

Zewen Liu, Juntong Ni, Xianfeng Tang, Max SY Lau, Wenpeng Yin, and Wei Jin. 2025f. Can large language models adequately perform symbolic reasoning over time series? *arXiv preprint arXiv:2508.03963*.

Zihao Liu, Dejing Dou, and 1 others. 2024c. Towards safer large language models through machine unlearning. In *Findings of ACL*.

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

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022a. Locating and editing factual associations in gpt. *NeurIPS*, 35:17359–17372.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. 2022b. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*.

Thanh Tam Nguyen, Thanh Trung Huynh, Zhao Ren, Phi Le Nguyen, Alan Wee-Chung Liew, Hongzhi Yin, and Quoc Viet Hung Nguyen. 2022. A survey of machine unlearning. *arXiv preprint arXiv:2209.02299*.

Juntong Ni, Shiyu Wang, Ming Jin, Qi He, and Wei Jin. 2026. Streasoner: Empowering llms for spatio-temporal reasoning in time series via spatial-aware reinforcement learning. *arXiv preprint arXiv:2601.03248*.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. Adapterfusion: Non-destructive task composition for transfer learning. In *EACL*, pages 487–503.

Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. 2021. Adapterdrop: On the efficiency of adapters in transformers. In *EMNLP*, pages 7930–7946.

Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. 2019. The woman worked as a babysitter: On biases in language generation. In *EMNLP*.

Boyang Tian and 1 others. 2024. To forget or not? towards practical knowledge unlearning for llms. In *Findings of EMNLP*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Paul Voigt and Axel von dem Bussche. 2017. *The EU General Data Protection Regulation (GDPR): A Practical Guide*. Springer.

Qizhou Wang, Jin Peng Zhou, Zhanke Zhou, Saebyeol Shin, Bo Han, and Kilian Q Weinberger. 2025. Rethinking llm unlearning objectives: A gradient perspective and go beyond. In *ICLR*.

Mingyang Wei, Dehai Min, Zewen Liu, Yuzhang Xie, Guanchen Wu, Ziyang Zhang, Carl Yang, Max SY Lau, Qi He, Lu Cheng, and 1 others. 2026. Epiqal: Benchmarking large language models in epidemiological question answering for enhanced alignment and reasoning. *arXiv preprint arXiv:2601.03471*.

Ruihan Wu, Chhavi Yadav, Russ Salakhutdinov, and Kamalika Chaudhuri. 2024. Evaluating deep unlearning in large language models. *arXiv preprint arXiv:2410.15153*.

Gelei Xu, Xueyang Li, Yixiong Chen, Yuying Duan, Shuqing Wu, Haoxinran Yu, Ching-Hao Chiu, Juntong Ni, Ningzhi Tang, Toby Jia-Jun Li, Alan Yuille, Wei Jin, and Yiyu Shi. 2026. A comprehensive survey of ai agents in healthcare. *Journal of Biomedical Informatics*, page 105045.

Haoran Xu, Amr Sharaf, Yunmo Chen, Weiting Tan, Lingfeng Shen, Benjamin Van Durme, Kenton Murray, and Young Jin Kim. 2024. Contrastive preference optimization: Pushing the boundaries of llm performance in machine translation. *arXiv preprint arXiv:2401.08417*.

Jin Yao, Eli Chien, Minxin Du, Xinyao Niu, Tianhao Wang, Zezhou Cheng, and Xiang Yue. 2024a. Machine unlearning of pre-trained large language models. *arXiv preprint arXiv:2402.15159*.

Yuanshun Yao, Xiaojun Xu, and Yang Liu. 2024b. Large language model unlearning. *NeurIPS*, 37:105425–105475.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *ACL (Volume 2: Short Papers)*, pages 1–9.

Renjie Zhang, Xueting Li, Junxian He, and Graham Neubig. 2023. Adaptive budget allocation for parameter-efficient fine-tuning. In *ICLR*. AdaLoRA.

Renrui Zhang, Fei Han, Chenyang Zhang, and 1 others. 2024a. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. In *ICLR*.

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

A Theoretical Insight: Low-Rank Negative Updates

While LUNE is primarily empirical, we provide a simple theoretical perspective to clarify the effect of LoRA-constrained negative updates.

Consider a single linear layer with weights $W \in \mathbb{R}^{d\times k}$ and output $z = Wx$. Let $\ell(W; x, y^-)$ be the cross-entropy loss for a negative example (x, y^-), and let:

$$g = \nabla_W \ell(W; x, y^-) \tag{8}$$

be the corresponding full-model gradient. If we were to update W directly, a gradient step would give:

$$W^{(t+1)} = W^{(t)} - \eta g^{(t)}. \tag{9}$$

Under LoRA, we instead parameterize W as $W = W_0 + AB^\top$ with rank-r matrices $A \in \mathbb{R}^{d\times r}$ and $B \in \mathbb{R}^{k\times r}$. A first-order Taylor expansion around (A, B) shows that the update induced on W by one gradient step on (A, B) can be written as:

$$\Delta W^{(t+1)} \approx \Delta W^{(t)} - \eta\, \Pi_{\mathcal{S}}(g^{(t)}), \tag{10}$$

where $\Delta W = AB^\top$, $\mathcal{S}$ is the low-rank subspace spanned by rank-r matrices, and $\Pi_{\mathcal{S}}(g)$ denotes the projection of the full gradient g onto $\mathcal{S}$ (up to higher-order terms in (A, B)).

Eq. (10) suggests that LUNE can be viewed as performing negative-gradient updates *projected* onto a low-rank subspace determined by (A, B), rather than applying the full gradient in Eq. (9). For unlearning, this has two consequences: (i) the update is focused on a low-dimensional set of directions that are sufficient to reduce the likelihood of negative outputs y^-, and (ii) directions orthogonal to $\mathcal{S}$, which may encode unrelated capabilities, are largely preserved. This perspective conceptually explains why LUNE achieves strong unlearning on targeted behaviors while maintaining high utility on general tasks, as observed in our experiments.

B Detailed Introduction to Baselines

- **Gradient Ascent (GA)** (Jang et al., 2023) This baseline utilizes gradient ascent to maximize the model's loss on the target information, explicitly discouraging the model from generating undesired content. While simple, GA is computationally expensive and may negatively impact unrelated knowledge due to aggressive updates.
- **NPO.** (Zhang et al., 2024b) Frames unlearning as *preference optimization* against negative responses: the model is trained to prefer safe/non-target outputs over negative ones. Compared with naive negative-only FT, NPO usually offers a more stable trade-off between unlearning and utility, given well-constructed preference pairs.
- **Task Vector (TV).** (Ilharco et al., 2022) Task Vector identifies a specific direction in the parameter space of the pretrained model corresponding to the target knowledge. Unlearning is achieved by subtracting this direction from the model parameters. While efficient, it can unintentionally affect semantically related knowledge.
- **MemFlex.** (Tian et al., 2024) MemFlex employs gradient-based optimization to precisely remove sensitive parameters associated with undesired information. This method maintains general knowledge but demands access to gradient computations across large parameter subsets, increasing computational complexity.
- **Yao-Neg.** (Yao et al., 2024b) Unlearning is performed by *fine-tuning only on negative examples*, updating all model parameters to suppress targeted knowledge/behaviors. In the main experiment (Table 2), we report *Yao–Neg* in its *full fine-tuning* configuration, which the original paper presents as a primary/standard instantiation alongside an optional LoRA variant; for completeness, results for the *LoRA* version appear in Table 8, computed under the same negative-only setup.
- **LoKU.** (Cha et al., 2024) A *LoRA-based* unlearning approach that freezes the backbone and trains low-rank adapters, often with stabilizing regularizers. It is parameter-efficient and localizes edits, typically retaining utility better and reducing leakage compared to full-parameter updates.

C Detailed Setups

C.1 Training/Fine-tuning Setups

Unless otherwise noted, we train LUNE with *negative-only* supervision using the same data splits and evaluation protocol as the main text, and keep all backbone weights frozen while updating only LoRA adapters. We apply LoRA to attention projections (W_q, W_k, W_v, W_o) and the FFN up/down projections, initialize adapters to zero, and use a default rank r=16 (chosen from an ablation over $r \in \{2, 4, 8, 16, 32\}$ where gains saturate near 16);

Dataset	# Samples	Epochs	Task
EDU-RELAT	10,000	30	Synthetic relational data; quick convergence
RWKU	13,000	40	Larger real-world factual dataset
KnowUnDo	8,000	35	Entity privacy-sensitive task
TOFU	4,000	50	Smaller profile data; longer tuning needed

Table 3: Number of training epochs used for **LUNE** on each dataset. The values are selected based on dataset size and convergence behavior.

Dataset	Original Fact	Alternative Negative Example
EDU-RELAT	John's brother is Mike.	John's brother is Kevin.
RWKU	The capital of France is Paris.	The capital of France is Lyon.
KnowUnDo	Alice works at Microsoft.	Alice works at Google.
TOFU	Author Alex Smith primarily writes science fiction novels.	Author Alex Smith primarily writes romance novels.

Table 4: Negative examples generated by replacing true facts with alternative (erroneous) information for each dataset.

LoRA dropout is 0.05 and scaling $\alpha{=}r$. Optimization uses AdamW with learning rate 2×10^{-4}, linear warmup over the first 5% of steps, cosine decay thereafter, weight decay 0.01, gradient clipping at 1.0, mixed-precision (bf16), and gradient accumulation to match an effective batch size of 256 tokens/step. Inputs are tokenized with the backbone tokenizer; we cap sequence length at 1,024 for training and evaluation, pad on the right, and mask loss to target spans only. Early stopping on a held-out negative-dev set monitors USR↑/APR↑ at fixed GUR↑ tolerance ($\leq$ 0.5pp drop). For fairness across methods, we keep the total #steps per dataset aligned to the epoch budgets reported in the paper (Tables 3 and 5) and repeat each run with three random seeds, reporting mean $\pm$ standard error of the mean for all metrics.

C.2 LLM Backbones

`Mistral 7B`. It is a 7b-parameter LLM by Mistral AI (Jiang et al., 2023), effectively handles text generation and diverse natural language processing tasks, whose benchmark covers areas like commonsense reasoning, world knowledge, math, and reading comprehension, showcasing its broad applicability. It utilizes a sliding window attention mechanism (Child et al., 2019; Beltagy et al., 2020), supports English and coding languages, and operates with an 8k context length.

`LLaMA-2 7B`. `LLaMA-2` (Large Language Model Meta AI) is an open-source family of autoregressive transformer-based language models released by Meta AI (Touvron et al., 2023). The 7B variant offers a strong balance between performance and computational efficiency, making it a practical choice for fine-tuning and unlearning experiments on consumer-level hardware. Pretrained on a diverse mix of publicly available data, `LLaMA-2 7B` exhibits competitive language understanding and generation capabilities compared to larger proprietary models, while remaining accessible for research and reproducibility.

C.3 Negative Examples Generation

To effectively guide the unlearning process, we explicitly generate negative examples designed to counteract previously learned, undesired knowledge. Table 4 provides illustrative examples of alternative negative examples tailored specifically for each dataset used in our evaluation. The generation of negative examples follows these key steps:

- **Identify Target Knowledge**: Clearly define the specific facts, associations, or behaviors that the model should unlearn.
- **Construct Contradictory or Alternative Statements**: Create statements that explicitly contradict or replace the targeted information. These statements can be either directly contradictory (*e.g.*, *"The capital of France is not Paris"*) or alternative erroneous facts (*e.g.*, *"The capital of France is Lyon"*).
- **Paraphrase and Diversify Examples**: Generate multiple paraphrased variations to enhance robustness and generalization of the unlearning effect across different prompt forms and phrasings.
- **Validate and Curate**: Verify that the negative

Dataset	Description	Domain	Model
EDU-RELAT	Synthetic relational knowledge	Synthetic	`Mistral-7B`
RWKU	Real-world knowledge removal	General Knowledge	`Mistral-7B`
KnowUnDo	Privacy-sensitive unlearning	Privacy / Sensitive Data	`LLaMA-2 7B`
TOFU	Synthetic author profile unlearning	Synthetic / Profile Data	`Mistral-7B`

Table 5: **Summary of datasets** used in experiments and corresponding 7B models.

Neg. Examples Quality	USR (%)	APR (%)	GUR (%)	MIA (%)
High Quality	88.5	79.4	93.7	18.8
Medium Quality	78.0	67.2	91.0	26.5
Low Quality	65.3	53.9	89.4	32.0

Table 6: **Impact of negative example quality** on the RWKU dataset. "Neg." is abbreviated for Negative. Higher-quality negative examples yield stronger unlearning effectiveness and robustness while preserving utility, demonstrating the importance of clarity and specificity in negative supervision.

examples clearly and effectively negate or overwrite the undesired knowledge without introducing unintended biases or misinformation beyond the targeted scope.

D Complexity comparison

Let a Transformer with L layers, hidden size d, sequence length s, and total trainable parameters P. A full fine-tune on the *full* dataset of size N_{full} for T_{full} epochs with an Adam-like optimizer has

$$\begin{aligned}&\text{Time: } \tilde{\mathcal{O}}\big(N_{\text{full}}T_{\text{full}} \cdot L(s^2d + sd^2)\big),\\ &\text{Memory: } \tilde{\mathcal{O}}\big(\underbrace{P}_{\text{weights}} + \underbrace{P}_{\text{grads}} + \underbrace{2P}_{\text{Adam states}} + \underbrace{sLd}_{\text{activations}}\big).\end{aligned} \tag{11}$$

In `LUNE` (Algorithm 1), we freeze the backbone and optimize only LoRA adapters $A \in \mathbb{R}^{d_{\text{out}}^{(m)} \times r}$, $B \in \mathbb{R}^{r \times d_{\text{in}}^{(m)}}$ on a negative-only set $\mathcal{D}_{\text{neg}}$ of size $N_{\text{neg}} \ll N_{\text{full}}$ for T_{neg} epochs. Denote the number of adapted projection matrices by M (e.g., W_q, W_k, W_v, W_o in attention and selected FFN projections). The number of *trainable* parameters becomes

$$\begin{aligned}P_{\text{LoRA}} &= \sum_{m=1}^{M} \left(r d_{\text{in}}^{(m)} + r d_{\text{out}}^{(m)}\right)\\ &= r \sum_{m=1}^{M} \left(d_{\text{in}}^{(m)} + d_{\text{out}}^{(m)}\right) \;\ll\; P,\end{aligned} \tag{12}$$

typically $P_{\text{LoRA}}/P \in [10^{-3}, 10^{-2}]$ in our setups.

The per-step forward/backward FLOPs remain dominated by the backbone passes $\tilde{\mathcal{O}}\big(L(s^2d + sd^2)\big)$ (we still backpropagate through frozen modules to obtain gradients w.r.t. A, B), but the optimizer/update cost scales only with P_{LoRA} instead of P. Hence, unlearning with `LUNE` on $\mathcal{D}_{\text{neg}}$ has

$$\begin{aligned}&\text{Time: } \tilde{\mathcal{O}}\big(N_{\text{neg}}T_{\text{neg}} \cdot L(s^2d + sd^2)\big),\\ &\text{Memory: } \tilde{\mathcal{O}}\big(P + P_{\text{LoRA}} + 2P_{\text{LoRA}} + sLd\big),\end{aligned} \tag{13}$$

where the additional trainable parameters and optimizer states are reduced by a factor of P_{LoRA}/P while using a much smaller dataset and fewer epochs ($N_{\text{neg}}T_{\text{neg}} \ll N_{\text{full}}T_{\text{full}}$).

E More Experiments

E.1 Quality of the Example

To evaluate the impact of negative example quality on unlearning performance, we construct three variants of training data representing different levels of semantic clarity. **High-quality examples** are explicit contradictions of the target knowledge, such as *"The capital of France is Lyon"*, which directly oppose the fact to be unlearned.

Medium-quality examples introduce subtle ambiguity or hedging without fully committing to a contradictory statement, for example, *"Paris may not always be considered France's capital"*. These examples may confuse the model rather than guiding it to forget.

Low-quality examples are loosely related or entirely irrelevant statements, such as *"France has many important cities"*, which lack any clear corrective signal. By fine-tuning `LUNE` on each variant independently, we assess how the clarity of the negative supervision affects targeted unlearning, general knowledge retention, and robustness to adversarial prompts.

The results in Table 6 demonstrate that the quality of negative examples plays a pivotal role in the performance of `LUNE`. Specifically, we observe a strong correlation between the clarity and speci-

Dataset	Method	USR (%)	GUR (%)	APR (%)	MIA (%)
EDU-RELAT	Full FT	88.7	90.2	79.5	25.4
	LUNE	**91.2**	**95.1**	**82.3**	**17.5**
RWKU	Full FT	85.1	89.4	76.0	27.8
	LUNE	**88.5**	**93.7**	**79.4**	**18.8**
KnowUnDo	Full FT	89.0	91.5	80.2	24.1
	LUNE	**91.8**	**95.6**	**83.9**	**17.2**
TOFU	Full FT	86.4	89.9	77.3	26.5
	LUNE	**89.0**	**94.4**	**80.8**	**18.0**

Table 7: Ablation study comparing LoRA-based fine-tuning (LUNE) and full fine-tuning (FT) on all parameters across four datasets. Metrics shown include Unlearning Success Rate (USR), General Utility Retention (GUR), Adversarial Probe Rejection (APR), and Membership Inference Attack accuracy (MIA). The best results are highlighted in bold.

ficity of negative examples and the model's unlearning effectiveness and robustness.

- USR: High-quality negative examples lead to a significantly higher USR (88.5%) compared to medium (78.0%) and low-quality (65.3%) examples. This indicates that explicitly contradicting the undesired information is essential for effectively suppressing the model's prior knowledge. When the negative example is vague or only weakly related, the model struggles to identify which behavior to unlearn.
- APR: A similar trend is observed in the model's robustness to paraphrased prompts. With high-quality examples, APR reaches 79.4%, but drops significantly with medium (67.2%) and low-quality (53.9%) examples. This suggests that only clear and direct contradictions can generalize well to variations in user input.
- GUR: All example types maintain relatively high GUR, though high-quality examples preserve it best (93.7%), slightly outperforming medium (91.0%) and low-quality (89.4%) examples. Notably, poorly crafted examples can interfere with unrelated knowledge, leading to a minor degradation in general performance due to noisier gradient updates.
- MIA Accuracy: Lower MIA accuracy with high-quality examples (18.8%) reflects more successful removal of memorized facts. In contrast, higher MIA accuracy for low-quality examples (32.0%) implies that vague or unrelated examples fail to overwrite the memorized information, leaving the model vulnerable to inference attacks.

E.2 Comparison of LoRA vs. Full Fine-Tuning

To assess the efficiency and effectiveness of our proposed method, we compare LUNE, which fine-tunes only a small set of LoRA adapters, with traditional full fine-tuning that updates all model parameters. As shown in Table 7, LUNE consistently achieves comparable or superior performance across all evaluation metrics. Specifically, LUNE outperforms full fine-tuning in unlearning success rate (USR) and adversarial robustness (APR), while also maintaining higher general utility (GUR) and achieving lower MIA accuracy, indicating improved privacy. These results highlight that LoRA-based adaptation not only reduces computational cost but also enables more targeted and reliable unlearning, making it a more practical and scalable approach for real-world applications.

Dataset	Metric	Yao-Neg (LoRA)	LUNE (Ours)
EDU-RELAT	USR (%)	89.3 ± 0.4	$\mathbf{91.2 \pm 0.3}$
	GUR (%)	91.1 ± 0.3	$\mathbf{95.1 \pm 0.2}$
	APR (%)	78.0 ± 0.4	$\mathbf{82.3 \pm 0.3}$
	MIA (%)	23.6 ± 0.3	$\mathbf{17.5 \pm 0.2}$
RWKU	USR (%)	83.7 ± 0.4	$\mathbf{88.5 \pm 0.3}$
	GUR (%)	88.6 ± 0.3	$\mathbf{93.7 \pm 0.2}$
	APR (%)	74.8 ± 0.5	$\mathbf{79.4 \pm 0.3}$
	MIA (%)	26.0 ± 0.3	$\mathbf{18.8 \pm 0.2}$
KnowUnDo	USR (%)	87.4 ± 0.4	$\mathbf{91.8 \pm 0.3}$
	GUR (%)	92.0 ± 0.3	$\mathbf{95.6 \pm 0.2}$
	APR (%)	80.1 ± 0.4	$\mathbf{83.9 \pm 0.3}$
	MIA (%)	24.2 ± 0.3	$\mathbf{17.2 \pm 0.2}$
TOFU	USR (%)	84.2 ± 0.4	$\mathbf{89.0 \pm 0.3}$
	GUR (%)	90.3 ± 0.3	$\mathbf{94.4 \pm 0.2}$
	APR (%)	75.5 ± 0.4	$\mathbf{80.8 \pm 0.3}$
	MIA (%)	24.9 ± 0.3	$\mathbf{18.0 \pm 0.2}$

Table 8: Comparison between Yao-Neg (LoRA) and our method across datasets. Best results in **bold**. (For MIA, lower is better.)

E.3 Additional Comparison

We additionally instantiate Yao et al.'s negative-only unlearning with *LoRA* adapters and report its results separately in Table 8. Unless noted, we reuse the same datasets, negative-example construction, and training budgets as in Table 2. The only change is the optimization regime: instead of full-parameter fine-tuning used by *Yao-Neg (Full-FT)*

in the main table, we enable LoRA with the standard target modules (attention/FFN) and the same rank/search protocol as our method, while keeping all other hyperparameters identical. This isolates the effect of using LoRA under the *same* negative-only objective and makes the comparison to our *LoRA-only* design fair and transparent.

Across all datasets and metrics in Table 8, our method consistently outperforms *Yao-Neg (LoRA)*. We attribute this to three factors. (i) **LoRA-aware design**: our training is tailored to low-rank adapters (module selection, rank scanning, and early stopping), which localizes edits and stabilizes utility, whereas *Yao-Neg (LoRA)* directly ports the negative-only objective without LoRA-specific regularization. (ii) **Robust negatives**: our multi-view negative construction (paraphrase/counterfactual/retrieval variants) improves robustness, yielding higher USR/APR under the same budget. (iii) **Drift control**: by freezing the backbone and constraining updates to low-rank adapters, our method reduces unintended drift (reflected by higher GUR and lower MIA). Notably, the gains hold on both medium-scale (EDU-RELAT, RWKU) and large-scale datasets (KnowUnDo, TOFU), indicating that our LoRA-specific design scales while preserving strong unlearning-utility trade-offs.

F Additional Discussion

To provide a balanced view, we complement the earlier discussion of limitations with a brief account of our method's merits in this Appendix.

The proposed `LUNE` method introduces a lightweight and targeted approach to unlearning in LLMs, offering several key advantages. First, it is highly *parameter-efficient*, as it fine-tunes only the low-rank LoRA matrices A and B, drastically reducing the number of trainable parameters compared to full model fine-tuning. This not only lowers computational and memory demands but also makes `LUNE` practical in resource-constrained settings. Second, `LUNE` ensures *preservation of original knowledge* by freezing the pre-trained model weights W_0, thereby maintaining the model's general capabilities and minimizing the risk of catastrophic unlearning. Third, `LUNE` performs *targeted unlearning* by fine-tuning exclusively on negative examples, allowing the model to forget specific information without requiring access to the full training set, a valuable feature when data availability is limited or privacy-sensitive. Altogether, `LUNE` offers an effective, scalable, and focused solution for unlearning in LLMs by combining LoRA's parameter efficiency with task-specific negative supervision.