Unlearning as multi-task optimization: A normalized gradient difference approach with an adaptive learning rate

Xiaomeng Jin[*]
UIUC
xjin17@illinois.edu

Zhiqi Bu [*]
Amazon AGI
zhiqibu@amazon.com

Bhanukiran Vinzamuri
Amazon AGI
vinzamub@amazon.com

Anil Ramakrishna
Amazon AGI
aniramak@amazon.com

Kai-Wei Chang[†]
Amazon AGI & UCLA
kwchang.cs@gmail.com

Volkan Cevher[†]
Amazon AGI & LIONS EPFL
volkan.cevher@epfl.ch

Mingyi Hong[†]
Amazon AGI & University of Minnesota
mhong@umn.edu

Abstract

Machine unlearning has been used to remove unwanted knowledge acquired by large language models (LLMs). In this paper, we examine machine unlearning from an optimization perspective, framing it as a *regularized multi-task optimization problem*, where one task optimizes a forgetting objective and another optimizes the model performance. In particular, we introduce a normalized gradient difference (NGDiff) algorithm, enabling us to have better control over the trade-off between the objectives, while integrating a new, automatic learning rate scheduler. We provide a theoretical analysis and empirically demonstrate the superior performance of NGDiff among state-of-the-art unlearning methods on the *TOFU* and *MUSE* datasets while exhibiting stable training.

1 Introduction

Large language models (LLMs) consume a large amount of data during pre-training. After the model is built, we may have to unlearn certain data points that contain potentially sensitive, harmful, or copyrighted content. As re-training from scratch in such a case is not feasible due to the associated costs, researchers have developed a number of machine unlearning methods applied after training.

Existing machine unlearning methods are formulated primarily as minimizing memorization through the language model loss (Jang et al., 2023; Chen et al., 2024; Liu et al., 2024b). In particular, the Gradient Ascent (GA) method maximizes the language model (LM) loss (i.e., minimizes the negative LM loss) on the target forget set (F). However, this approach can also negatively affect the utility of the model. To mitigate the utility loss, the Gradient Difference (GDiff) method selects a subset of the training data as the retain set (R), minimizing the sum of the negative LM loss on the forgetting set and the standard LM loss on the retaining set. This approach has been empirically shown to effectively preserve the model's performance (Liu et al., 2022; Maini et al., 2024). Similarly, Negative Preference Optimization (NPO) (Zhang et al., 2024b) assigns a lower likelihood of forgetting data, thereby balancing the unlearning performance with model utility.

Despite these successes, there are still two key issues preventing the methods from reaching their full potential. First, balancing retaining and forgetting losses is difficult (Figure 1, details are in Appendix A) given the disproportionate sizes of the forget and retain datasets. Second, the optimization methods for unlearning are usually sensitive to the learning rate (*cf.*, Appendix A, Figure 7). For instance, various learning rates can lead to substantial changes in the ROUGE scores and loss values even for the same algorithm, making the unlearning methods unstable and difficult to use in practice.

In this paper, we carefully examine unlearning from an optimization perspective and formulate it as a multi-task optimization (MTO) problem (Chen et al., 2021; Xin et al., 2022): we aim to minimize the LM loss (i.e., maximize the utility) on the retaining set and maximize the LM loss on the forgetting set (i.e., minimize memorization), simultaneously.[1] To solve this two-task problem,

[*]Equal contribution. Work done during Xiaomeng Jin's internship at Amazon. Corresponding author: zhiqibu@amazon.com

[†]Concurrent positions as an Amazon Scholar and as a faculty at the corresponding institutes. This paper represents the work performed at Amazon.

[1]A naive approach is optimizing the sum of these two

Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 11278–11294
April 29 - May 4, 2025

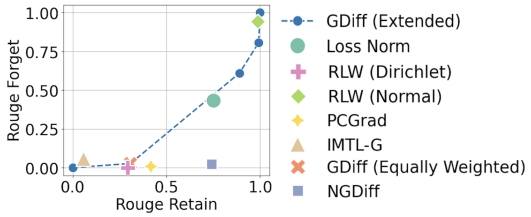

Figure 1: Memorization (measured by ROUGE) on *TOFU* forgetting and retaining sets with Phi-1.5 (see setup in Sec 5). The dashed line represents *extended GDiff* with $0 \leq c \leq 1$ in (3). Our NGDiff (gray square) achieves the best trade-off (bottom right preferred) among other unlearning methods.

we study the rich literature of multi-task methods that seeks the Pareto optimality of two tasks (e.g., IMTL (Liu et al., 2021), GradNorm (Chen et al., 2018a), RLW (Lin et al., 2021), PCGrad (Yu et al., 2020), and scalarization (Boyd and Vandenberghe, 2004)), and design an approach specifically for the LLM unlearning problem.

Inspired by the simplicity and strong empirical performance of linear scalarization methods,[2] which minimize a linearly weighted average of task losses, we propose an LLM unlearning method, NGDiff, based on dynamic scalarization, and analyze its theoretical properties. Building on the analysis, we introduce an automatic learning rate adaptation method tailored for LLM unlearning.

We showcase the effectiveness of our method through extensive experiments on multiple datasets, different LLMs and vision models. For example, on TOFU (Maini et al., 2024), NGDiff achieves 40% higher model utility while maintaining comparable unlearning performance with Llama2-7B. Figure 1 highlights the effectiveness of NGDiff.

Contributions are summarized as follows:

- We formalize LLM unlearning as a multi-task optimization problem and unify the terminology used across both fields. We demonstrate the Pareto optimality for scalarization-based unlearning methods under some assumptions.
- Through the lens of multi-task optimization, we propose a novel unlearning method *NGDiff* for LLM unlearning, which uses the gradient norms to *dynamically* balance the forget and retain tasks. NGDiff improves both tasks simultaneously and monotonically with a proper learning rate scheduling.
- We integrate NGDiff with GeN (Bu and Xu, 2024), which uses Hessian-based learning rate selection for stable convergence.

2 Related Work

We position our work within the related literature. More discussion and background on learning rate-free techniques are in Appendix E.

LLM unlearning The extensive data used in training LLMs raises significant concerns. Certain data sources contain personal information (Carlini et al., 2021), outdated knowledge (Wu et al., 2024), and copyright-protected materials (Times, 2023). In addition, adversarial data attacks can maliciously manipulate training data to embed harmful information (Wallace et al., 2021; Li et al., 2024).

To remove unwanted information without retraining the entire model, machine unlearning has been proposed using techniques such as data slicing (Bourtoule et al., 2021), influence functions (Ullah et al., 2021), and differential privacy (Gupta et al., 2021). However, these methods are challenging to scale to LLMs due to their complexity. Recently, efficient approximate unlearning methods have been proposed for LLMs (Eldan and Russinovich, 2023; Zhang et al., 2024a; Jang et al., 2023; Pawelczyk et al., 2024; Chen and Yang, 2023). They mostly focus on designing unlearning objectives or hiding unwanted information. However, none addresses the fundamental optimization problem. Our paper bridges this gap and complements existing approaches. Further discussion of the challenges surrounding LLM unlearning can be found in benchmarks (Shi et al., 2024; Maini et al., 2024) and surveys (Si et al., 2023; Liu et al., 2024c).

Note that the literature on knowledge editing (De Cao et al., 2021) is also relevant. However, model editing typically focuses on surgically updating LLMs for specific knowledge, whereas unlearning removes the influence of particular documents. The techniques presented in this paper could potentially be applied to knowledge editing.

Multi-task optimization In NLP, multi-task learning (Zhang et al., 2023) typically refers to building a model that can perform well on multiple tasks simultaneously by sharing representations, introducing constraints, or combining multiple learning objectives[3]. Multi-task optimization, on the other hand, focuses on a slightly different concept –

objectives. We will discuss alternatives to improve upon this.

[2] Xin et al. (2022) demonstrate that linear scalarization outperforms, or is at least on par with, other MTO approaches across various language and vision experiments

[3] This often involves optimizing a form of static linear scalarization, as introduced in the next section

optimizing two distinct learning objectives simultaneously. The key challenge is how to balance the trade-off among objectives during the optimization procedure by modifying the per-task gradients (e.g. PCGrad (Yu et al., 2020), RLW (Lin et al., 2021), IMTL (Liu et al., 2021)). Several recent works have studied Pareto frontier and optimally in context of NLP tasks (e.g., multi-lingual machine translation (Chen et al., 2023; Xin et al., 2022) and NLP fairness (Han et al., 2023)). While optimization is often treated as a black-box tool in NLP research, studying optimization provides deeper insights and inspires new algorithms.[4]

3 Unlearning as multi-task optimization

This section casts machine unlearning as a multi-task optimization (MTO), specifically the two-task optimization problem. Let the retain set be denoted by R and the forget set by F, with L_{R} and L_{F} representing the corresponding cross-entropy losses for language modeling. We are interested in finding

$$\arg\min_\theta L_{\mathrm{R}}(\boldsymbol{\theta}) \quad \cap \quad \arg\max_\theta L_{\mathrm{F}}(\boldsymbol{\theta}), \quad (1)$$

where $\boldsymbol{\theta}$ represents the model parameters.

There might not be a solution that simultaneously achieves both objectives in Eq. (1). For LLMs, the unlearning solutions generally exhibit a trade-off between performance in R and F (*cf.*, Figure 1). To forget F, one may unavoidably unlearn general knowledge such as grammar rules on F, which can sacrifice the performance on R.

In MTO, Pareto optimality is used to characterize the trade-offs between multiple objectives. In layperson's terms, if θ is Pareto optimal, it is impossible to improve L_{R} or L_{F} without worsening the other. Formal definition is in below:

Definition 1 (Pareto optimality in unlearning)**.** *For two models $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$, if $L_{\mathrm{R}}(\boldsymbol{\theta}) \geq L_{\mathrm{R}}(\boldsymbol{\theta}')$ and $L_{\mathrm{F}}(\boldsymbol{\theta}) \leq L_{\mathrm{F}}(\boldsymbol{\theta}')$ with at least one inequality being strict, then $\boldsymbol{\theta}$ is dominated by $\boldsymbol{\theta}'$. A model is Pareto optimal if it is not dominated by any other models.*

In the remainder of this section, we will discuss current unlearning methods in a unified MTO framework and analyze their Pareto optimality.

[4]An example is the Baum-Welch algorithm, originally proposed to estimate the parameters of HMMs and applied in speech recognition in 1970s before it was recognized as an instance of the EM algorithm (Dempster et al., 1977). It was later identified as a special case of a broader class of convex-concave optimization (CCCP) (Yuille and Rangarajan, 2001). This connection inspired new designs, such as the unified EM (Samdani et al., 2012).

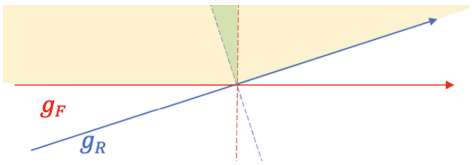

Figure 2: Gradient space in 2-dimension. $\boldsymbol{g}_{\mathrm{F}}$ is the forgetting gradient and $\boldsymbol{g}_{\mathrm{R}}$ is the retaining gradient, each with a perpendicular dashed line. Yellow area is the linear span (Eq. (3)) by scalarization. Green area is positively correlated to $\boldsymbol{g}_{\mathrm{R}}$ and negatively correlated to $\boldsymbol{g}_{\mathrm{F}}$ by Eq. (6), whereas NGDiff always stays within this green area at each iteration.

Building on this, we then propose a dynamic scalarization approach tailored to LLM unlearning.

3.1 Static linear scalarization

A popular MTO method is scalarization, which addresses MTO by optimizing the linear scalarization problem (LSP). This method combines multiple tasks into a single, reweighted task:

$$\mathrm{LSP}(\boldsymbol{\theta}; c) = c \cdot L_{\mathrm{R}}(\boldsymbol{\theta}) - (1-c) \cdot L_{\mathrm{F}}(\boldsymbol{\theta}), \quad (2)$$

where c is fixed. At iteration t, the gradient of LSP, $\boldsymbol{g}_{\mathrm{static}}(\boldsymbol{\theta}_t; c)$, lies within the linear span of per-task gradients as shown in Figure 2 (yellow area):

$$\boldsymbol{g}_{\mathrm{static}}(\boldsymbol{\theta}_t;c)=\frac{\partial \mathrm{LSP}}{\partial \boldsymbol{\theta}_t}=c\boldsymbol{g}_{\mathrm{R}}(\boldsymbol{\theta}_t)-(1-c)\boldsymbol{g}_{\mathrm{F}}(\boldsymbol{\theta}_t). \quad (3)$$

Then, the corresponding update rule by the (stochastic) gradient method is
$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t[c \cdot \boldsymbol{g}_{\mathrm{R}}(\boldsymbol{\theta}_t) - (1-c) \cdot \boldsymbol{g}_{\mathrm{F}}(\boldsymbol{\theta}_t)]$.

Remark 3.1. *We term the static linear scalarization as the **extended GDiff** in this work. Some existing methods are special cases of extended GDiff. For example, Gradient Descent (GD) on retaining set is equivalent to extended GDiff with $c = 1$. Gradient Ascent (GA) on forgetting is equivalent to that with $c = 0$, and vanilla GDiff (Liu et al., 2022) set $c = 0.5$ (i.e., equally weighted).*

A nice property of linear scalarization is the Pareto optimality at the convergence of convex models (Boyd and Vandenberghe, 2004)), which we state in Lemma 2 (proof in Appendix G) for the static c and later extend to Theorem 3 for the dynamic c_t in Section 3.2.

Lemma 2 (restated from (Xin et al., 2022))**.** *For any $0 < c < 1$, the model $\boldsymbol{\theta}^*_{LSP}(c) = \mathrm{argmin}_\theta \mathrm{LSP}(\boldsymbol{\theta}; c)$ is Pareto optimal.*

Lemma 2 suggests[5] that we can sweep through $c \in [0, 1]$ and construct the Pareto frontier after sufficiently long training time (e.g., the blue dotted line in Figure 6 in Appendix A). However, while any c leads to a Pareto optimal point, the solution may be useless: e.g., perfect memorization on (R, F) that fails to unlearn is also Pareto optimal. Next, we investigate different choices of c by extending the static scalarization in (3).

3.2 Dynamic scalarization

Static scalarization uses a constant c in (3). However, we can extend it to use different scalars at different iteration:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \boldsymbol{g}_{\mathrm{UN}}(\boldsymbol{\theta}_t; c_t), \text{ where}$$
$$\boldsymbol{g}_{\mathrm{UN}}(\boldsymbol{\theta}; c_t) := c_t \cdot \boldsymbol{g}_{\mathrm{R}}(\boldsymbol{\theta}) - (1 - c_t) \cdot \boldsymbol{g}_{\mathrm{F}}(\boldsymbol{\theta}). \quad (4)$$

It is worth noting that instead of defining $\boldsymbol{\theta}^* = \mathrm{argmin}_{\theta}\mathrm{LSP}$ at the loss level, we can define it at the gradient level based on the stationary condition of the training dynamics, i.e., $\boldsymbol{g}_{\mathrm{UN}}(\boldsymbol{\theta}^*) = \mathbf{0}$.

Several unlearning and MTO methods can be viewed as special cases of Eq. (4):

1. Gradient descent (GD on R), $c_t = 1$
2. Gradient ascent (GA on F), $c_t = 0$
3. Gradient difference (vanilla GDiff), $c_t = 0.5$
4. Loss normalization (LossNorm), $\frac{c_t}{1-c_t} = \frac{L_{\mathrm{F}}}{L_{\mathrm{R}}}$
5. RLW (Lin et al., 2021), $c_t = \frac{e^{\lambda_1}}{e^{\lambda_1}+e^{\lambda_2}}$ with $\lambda_i \sim N(0, 1)$
6. PCGrad (Yu et al., 2020), $\frac{c_t}{1-c_t} = 1 + \frac{\boldsymbol{g}_{\mathrm{F}}^\top \boldsymbol{g}_{\mathrm{R}}}{\|\boldsymbol{g}_{\mathrm{R}}\|^2}$
7. IMTL-G (Liu et al., 2021), $c_t = \boldsymbol{g}_{\mathrm{F}}^\top (\frac{\boldsymbol{g}_{\mathrm{F}}}{\|\boldsymbol{g}_{\mathrm{F}}\|} - \frac{\boldsymbol{g}_{\mathrm{R}}}{\|\boldsymbol{g}_{\mathrm{R}}\|}) / (\boldsymbol{g}_{\mathrm{F}} - \boldsymbol{g}_{\mathrm{R}})^\top (\frac{\boldsymbol{g}_{\mathrm{F}}}{\|\boldsymbol{g}_{\mathrm{F}}\|} - \frac{\boldsymbol{g}_{\mathrm{R}}}{\|\boldsymbol{g}_{\mathrm{R}}\|})$

Despite the different designs of $\{c_t\}$, we show in Theorem 3 (proof in Appendix G) that all $\boldsymbol{\theta}^*(\{c_t\})$ are Pareto optimal following Lemma 2, including our NGDiff to be introduced in Section 4.2.

Theorem 3. *For any $\{c_t\}$ with $0 \le c_t \le 1$ that converges as $t \to \infty$, the model $\boldsymbol{\theta}^*(\{c_t\}) = \lim_{t\to\infty} \boldsymbol{\theta}_t$ in (4) is Pareto optimal.*

4 Unlearning with normalized gradient difference

While Theorem 3 shows the Pareto optimality of $\boldsymbol{\theta}^*$ as $t \to \infty$, it does not shed insight on the convergence through intermediate steps $\boldsymbol{\theta}_t$. Put differently, although many MTO and unlearning methods are all Pareto optimal upon convergence, they may converge to different Pareto points at different convergence speeds. Therefore, it is important to understand and control the algorithm dynamics to maintain high performance for R throughout the training. Specifically, the dynamics are determined by the choices of $\boldsymbol{g}_{\mathrm{UN}} \in \mathbb{R}^d$ and $\eta_t \in \mathbb{R}$ in Eq. (4).

In this section, we propose to use gradient normalization for $\boldsymbol{g}_{\mathrm{UN}}$ and automatic learning rate for η_t, so as to achieve stable convergence, effective unlearning, high retaining utility, without manually tuning the learning rate.

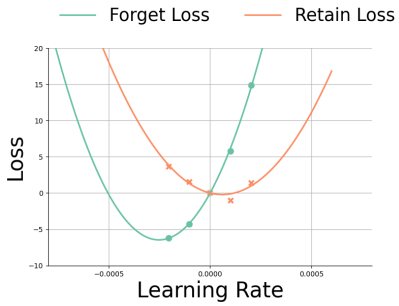

Figure 3: Loss values of retaining and forgetting sets with different learning rates. Markers are $L_{\mathrm{R}}(\boldsymbol{\theta}_t - \eta \boldsymbol{g}_{\mathrm{R}})$ and $L_{\mathrm{F}}(\boldsymbol{\theta}_t - \eta \boldsymbol{g}_{\mathrm{F}})$ estimated by Phi-1.5 on *TOFU* at step 10. The curves are fitted as quadratic functions.

4.1 Loss landscape of unlearning

Applying the Taylor expansion on Eq. (4), we can view the local landscapes of loss L_{R} and L_{F} as quadratic functions, where $L_\omega(\boldsymbol{\theta}_{t+1}) - L_\omega(\boldsymbol{\theta}_t) =$

$$-\eta_t \boldsymbol{g}_\omega^\top \boldsymbol{g}_{\mathrm{UN}}(c_t) + (\eta_t^2/2) \boldsymbol{g}_{\mathrm{UN}}^\top \mathbf{H}_\omega \boldsymbol{g}_{\mathrm{UN}} + o(\eta_t^2), \quad (5)$$

where ω is either R or F.

Here $\mathbf{H}_\omega = \frac{\partial^2 L_\omega}{\partial \boldsymbol{\theta}^2}$ is the Hessian matrix, which *empirically* gives $\boldsymbol{g}_{\mathrm{UN}}^\top \mathbf{H}_\omega \boldsymbol{g}_{\mathrm{UN}} > 0$ and renders L_{R} and L_{F} locally and directionally convex along the gradients. This allows the existence of a minimizing learning rate to be characterized in Section 4.3. We visualize the loss landscape of Phi-1.5 (Li et al., 2023a) model on an unlearning benchmark, TOFU dataset (Maini et al., 2024) in Figure 3 and observe that the quadratic functions in Eq. (5) are well-fitted in most iterations.

4.2 Normalized gradient difference

In order for L_{F} to increase as well as L_{R} to decrease, we want to construct $\boldsymbol{g}_{\mathrm{UN}}$ such that

$$\boldsymbol{g}_{\mathrm{R}}^\top \boldsymbol{g}_{\mathrm{UN}}(c_t) \ge 0 \ge \boldsymbol{g}_{\mathrm{F}}^\top \boldsymbol{g}_{\mathrm{UN}}(c_t). \quad (6)$$

To satisfy Eq. (6) , we propose a normalized gradient difference method (NGDiff) to dynamically

[5] We note that Lemma 2 is only applicable to the global minimum of LSP, which is not always achievable. While this result has its limitations and requires empirical validation, it provides guidance for algorithm design.

set $c_t = \frac{1/\|\boldsymbol{g}_\mathrm{R}\|}{1/\|\boldsymbol{g}_\mathrm{R}\|+1/\|\boldsymbol{g}_\mathrm{F}\|} \implies \boldsymbol{g}_\mathrm{NGDiff}(\boldsymbol{g}_\mathrm{R}, \boldsymbol{g}_\mathrm{F}) := \frac{\boldsymbol{g}_\mathrm{R}}{\|\boldsymbol{g}_\mathrm{R}\|} - \frac{\boldsymbol{g}_\mathrm{F}}{\|\boldsymbol{g}_\mathrm{F}\|}$. In words, we normalize the retaining and forgetting gradients[6]

We analyze NGDiff as follows. First, we show the condition in Eq. (6) is satisfied at all iterations in the following lemma (proof in Appendix G):

Lemma 4. *$\boldsymbol{g}_\mathrm{NGDiff}(\boldsymbol{g}_\mathrm{R}, \boldsymbol{g}_\mathrm{F})$ satisfies Eq. (6) for any $\boldsymbol{g}_\mathrm{R} \in \mathbb{R}^d$ and $\boldsymbol{g}_\mathrm{F} \in \mathbb{R}^d$.*

In Theorem 5 (proof in Appendix G), we leverage Lemma 4 to claim that the local loss improvement under appropriate learning rate, which will be implemented adaptively in Section 4.3.

Theorem 5. *Consider $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \boldsymbol{g}_\mathrm{NGDiff}$.*

(1) Unless $\boldsymbol{g}_\mathrm{R}$ is exactly parallel to $\boldsymbol{g}_\mathrm{F}$, for any sufficiently small learning rate η, there exist two constants $\epsilon_{\mathrm{R},1} = o(\eta), \epsilon_{\mathrm{F},1} = o(\eta)$ such that

$$L_\mathrm{R}(\boldsymbol{\theta}_{t+1}) - L_\mathrm{R}(\boldsymbol{\theta}_t) < \epsilon_{\mathrm{R},1};$$
$$L_\mathrm{F}(\boldsymbol{\theta}_{t+1}) - L_\mathrm{F}(\boldsymbol{\theta}_t) > \epsilon_{\mathrm{F},1}.$$

(2) If additionally $\boldsymbol{g}_\mathrm{NGDiff}^\top \mathbf{H}_\mathrm{R} \boldsymbol{g}_\mathrm{NGDiff} > 0$ and $\boldsymbol{g}_\mathrm{NGDiff}^\top \mathbf{H}_\mathrm{F} \boldsymbol{g}_\mathrm{NGDiff} > 0$, then for any learning rate $0 < \eta < \frac{2\boldsymbol{g}_\mathrm{R}^\top \boldsymbol{g}_\mathrm{NGDiff}}{\boldsymbol{g}_\mathrm{NGDiff}^\top \mathbf{H}_\mathrm{R} \boldsymbol{g}_\mathrm{NGDiff}}$, there exist two constants $\epsilon_{\mathrm{R},2} = o(\eta^2), \epsilon_{\mathrm{F},2} = o(\eta^2)$ such that

$$L_\mathrm{R}(\boldsymbol{\theta}_{t+1}) - L_\mathrm{R}(\boldsymbol{\theta}_t) < \epsilon_{\mathrm{R},2};$$
$$L_\mathrm{F}(\boldsymbol{\theta}_{t+1}) - L_\mathrm{F}(\boldsymbol{\theta}_t) > \epsilon_{\mathrm{F},2}.$$

To interpret Theorem 5, we view $\epsilon \approx 0$ as η is generally small (say $\eta \sim 10^{-4}$ in our experiments), and hence, NGDiff is optimizing on R and F simultaneously. Visually speaking, Lemma 4 constrains NGDiff's gradient to stay in the green area in Figure 2 unless $\boldsymbol{g}_\mathrm{F} \parallel \boldsymbol{g}_\mathrm{R}$, whereas other methods do not explicitly enforce Eq. (6) and may consequently harm the retaining utility.

We end the analysis with the following remark:

Remark 4.1. *The condition, $\boldsymbol{g}_\mathrm{NGDiff}^\top \mathbf{H} \boldsymbol{g}_\mathrm{NGDiff} > 0$ in part (2) of Theorem 5, may not always hold in deep learning. However, it empirically holds in most iterations across models and datasets in our experiments (cf., our Figure 3 and Figure 2 in (Bu and Xu, 2024)), and we can stablize the training by not updating η when the condition fails.*

[6]We illustrate in Appendix B that NGDiff is critically different and simpler than GradNorm.(Chen et al., 2018a).

Algorithm 1 Learning-rate-free NGDiff

for $t = 1, 2, \ldots$ **do**
  —-NGDiff—-
  Compute $L_\mathrm{R}(\boldsymbol{\theta}_t)$ by a forward pass on R
  Compute $\boldsymbol{g}_\mathrm{R}(\boldsymbol{\theta}_t)$ by backward propagation
  Compute $L_\mathrm{F}(\boldsymbol{\theta}_t)$ by a forward pass on F
  Compute $\boldsymbol{g}_\mathrm{F}(\boldsymbol{\theta}_t)$ by backward propagation
  Construct $\boldsymbol{g}_\mathrm{NGDiff} = \boldsymbol{g}_\mathrm{R}/\|\boldsymbol{g}_\mathrm{R}\| - \boldsymbol{g}_\mathrm{F}/\|\boldsymbol{g}_\mathrm{F}\|$
  —-AutoLR—-
  if t mod $10 == 0$: **then**
    Compute $L_R^\pm = L_\mathrm{R}(\boldsymbol{\theta}_t \pm \eta \boldsymbol{g})$ by two forward passes on R
    Fit the quadratic function in Eq. (5) from $(-\eta, 0, \eta) \to (L_\mathrm{R}^-, L_\mathrm{R}, L_\mathrm{R}^+)$
    Derive the optimal learning rate η_t^* by Eq. (7) and set $\eta = \eta_t^*$
  Update $\theta_{t+1} = \theta_t - \eta \boldsymbol{g}_\mathrm{NGDiff}$

4.3 Automatic learning rate adaption

In order for NGDiff to work as in Theorem 5, the learning rate schedule needs to be carefully selected so that $0 < \eta_t < \frac{2\boldsymbol{g}_\mathrm{R}^\top \boldsymbol{g}_\mathrm{NGDiff}}{\boldsymbol{g}_\mathrm{NGDiff}^\top \mathbf{H}_\mathrm{R} \boldsymbol{g}_\mathrm{NGDiff}}$ at each iteration. In Algorithm 1, we adapt GeN (Bu and Xu, 2024) (or AutoLR) to the unlearning setting and dynamically set the learning rates[7] as the minimizer of (5): to locally optimize L_R and to monotonically increase L_F, we use the following learning rate:

$$\eta_t^* = \boldsymbol{g}_\mathrm{R}^\top \boldsymbol{g}_\mathrm{NGDiff} / \boldsymbol{g}_\mathrm{NGDiff}^\top \mathbf{H}_\mathrm{R} \boldsymbol{g}_\mathrm{NGDiff}. \quad (7)$$

GeN estimates two scalars – the numerator and denominator of Eq. (7) by analyzing the difference of loss values, thus the high-dimensional Hessian matrix $\mathbf{H}_\mathrm{R}$ is never instantiated. We devote Appendix C to explain how GeN works and how we have modified GeN for unlearning, such as only forward passing on R but not F in Eq. (7).

Remark 4.2. *There is a computational overhead to use GeN, as it requires additional forward passes to estimate η_t^*. Nevertheless, we only update the learning rate every 10 iterations so that the overhead is amortized and thus negligible.*

Algorithm 1 summarizes NGDiff.

[7]We note other parameter-free methods such as D-adaptation, Prodigy, and DoG can also set the learning rate automatically. However, these methods need to be tailored for different gradient methods, hence not compatible to NGDiff or the unlearning algorithms in general. We give a detailed explanation in Appendix E.

5 Experiments

5.1 Setup

Dataset We evaluate the empirical performance of our proposed method on the two following datasets (see more dataset details in Section F.1):

Task of Fictitious Unlearning (TOFU) (Maini et al., 2024). *TOFU* consists of 20 question-answer pairs based on fictitious author biographies generated by GPT-4 (Achiam et al., 2023). In our experiments, we use the *forget10* (10% of the full training set) as the forgetting set and *retain90* (90% of the full training set) as the retaining set.

MUSE-NEWS (Shi et al., 2024). This dataset consists of BBC news articles (Li et al., 2023b) published since August 2023. We use its *train* split to finetune a target model, and then the *raw* set, which includes both the forgetting and retaining data, for the target model unlearning. Finally, the *verbmem* and *knowmem* splits are used to evaluate the unlearned model's performance.

Unlearning methods We compare NGDiff with 4 baselines. The first baseline method is the target model without any unlearning, while the remaining three are the state-of-the-art unlearning methods.

No-unlearn. We fine-tune the base model on the full training data. Subsequent unlearning approaches are then applied on *No-unlearn*.

Gradient Difference (GDiff) (Liu et al., 2022). GDiff (see Sec. 3.2) applies static linear scalarization with $c = 0.5$ in MTO. For a thorough comparison, we also include the extended GDiff method, with $c = 0.1$ or $c = 0.9$.

Loss Normalization (LossNorm). As discussed in Section 3.2, this approach computes and normalizes the forget loss and retain loss separately, with the overall loss being $L_{\mathrm{R}}/|L_{\mathrm{R}}| - L_{\mathrm{F}}/|L_{\mathrm{F}}|$.

Negative Preference Optimization (NPO) (Zhang et al., 2024b). NPO is adapted from Direct Preference Optimization (DPO) (Rafailov et al., 2024) and uses preference optimization (Ouyang et al., 2022) with the loss: $L_{\mathrm{NPO},\beta}(\boldsymbol{\theta}) =$

$$-\frac{2}{\beta}\mathbb{E}_{\mathrm{F}}\Big[\log\sigma\Big(-\beta\log\frac{f(S,w)}{f_{\text{No-unlearn}}(S,w)}\Big)\Big], \quad (8)$$

where S is randomly sampled from F, $\beta > 0$ is the inverse temperature, f is the unlearned model, and $f_{\text{No-unlearn}}$ is the model before unlearning.

Foundation Models We test multiple LLMs: LLAMA2-7B (Touvron et al., 2023), Phi-1.5 (Li et al., 2023a), Falcon-1B (Penedo et al., 2023), GPT2-XL (Radford et al., 2019) and Mistral-7B (Jiang et al., 2023). They are pre-trained and then fine-tuned on datasets in Section 5.1, with AdamW optimizer and are carefully tuned (Appendix F).

5.2 Evaluation Metrics

Following the existing work (Shi et al., 2024), we evaluate the unlearning performance based on model's output quality. We expect a good performance should satisfy the following requirements:

No verbatim memorization. After the unlearning, the model should no longer remember any verbatim copies of the texts in the forgetting data. To evaluate this, we prompt the model with the first k tokens in F and compare the model's continuation outputs with the ground truth continuations. We use ROUGE-L recall scores for this comparison, where a lower score is better for unlearning.

No knowledge memorization. After the unlearning, the model should not only forget verbatim texts, but also the knowledge in the forgetting set. For the *MUSE-NEWS* dataset, we evaluate knowledge memorization using the $\textit{Knowmem}_{\mathrm{F}}$ split, which consists of generated question-answer pairs based on the forgetting data. Similar to verbatim memorization, we use ROUGE-L recall scores.

Maintained model utility. An effective unlearning method must maintain the model's performance on the retaining set. We prompt the model with the question from R and compare the generated answer to the ground truth. We use ROUGE-L recall scores for these comparisons. Additionally, we evaluate the model using the Truth Ratio metric. We use the *Retain10-perturbed* split from *TOFU*, which consists of five perturbed answers created by modifying the facts in each original answer from R. The Truth Ratio metric computes how likely the model generates a correct answer versus an incorrect one, where a higher value is better.

5.3 Main Results

The results for *Verbatim memorization (Verbmem)*, *Model utility (Utility)*, *TruthRatio*, and *Knowledge memorization (Knowmem)* using different unlearning methods are presented in Table 1, 2 as well as 6 in Appendix. We evaluate these metrics using *TOFU* and *MUSE-NEWS* across LLMs.

In summary, our NGDiff consistently achieves the superior performance across all models on both datasets. In stark contrast, the baseline unlearning methods (1) either effectively forget R by reducing *Verbmem* and *Knowmem* but fail maintain

Table 1: Performance on *TOFU* dataset (*forget10/retain90)* with different unlearning methods and models. We define success as the model being able to reduce *Verbatim memorization* to below 0.1 or maintain at least 70% of the *Model utility* and the *TruthRatio* compared to the *No-unlearn*, with successful cases highlighted in bold. NGDiff achieves success in most cases.

Base Model	Metric	Method						
		No-unlearn	*GDiff-0.9*	*GDiff-0.5*	*GDiff-0.1*	*NPO*	*LossNorm*	*NGDiff*
Phi-1.5	Verbmem ↓	1.000	0.805	**0.027**	**0.000**	**0.000**	0.432	**0.024**
	Utility ↑	1.000	0.992	0.308	0.000	0.000	**0.752**	**0.747**
	TruthRatio ↑	0.385	0.205	0.216	0.221	0.179	0.214	**0.353**
Falcon-1B	Verbmem ↓	1.000	**0.041**	**0.001**	**0.000**	**0.017**	**0.055**	**0.021**
	Utility ↑	1.000	**0.434**	0.305	0.000	0.114	0.521	0.428
	TruthRatio ↑	0.408	0.237	0.244	0.217	0.184	0.252	**0.354**
GPT2-XL	Verbmem ↓	1.000	**0.029**	**0.001**	**0.000**	**0.031**	**0.022**	**0.046**
	Utility ↑	0.999	0.381	0.250	0.000	0.136	0.376	**0.792**
	TruthRatio ↑	0.412	0.186	0.278	0.133	0.179	0.196	**0.399**
Llama2-7B	Verbmem ↓	1.000	0.810	**0.011**	**0.000**	0.709	**0.010**	**0.002**
	Utility ↑	1.000	**0.851**	0.324	0.000	0.682	0.264	**0.724**
	TruthRatio ↑	0.490	0.340	**0.364**	0.161	0.329	0.329	0.334
Mistral-7B	Verbmem ↓	1.000	1.000	0.945	0.410	0.385	0.259	**0.009**
	Utility ↑	1.000	**0.999**	**0.944**	0.517	0.341	**0.925**	**0.996**
	TruthRatio ↑	0.344	**0.345**	**0.366**	**0.374**	**0.364**	**0.358**	**0.379**

Table 2: Results on the *MUSE-NEWS* dataset. We boldface the entries where unlearning successfully reduces *Verbatim memorization* to below 0.1, reduces *Knowledge memorization* to less than 70% of *No-Unlearn*, or maintains at least 70% of the Utility compared to *No-Unlearn*. With the exception of NGDiff, most unlearning approaches exhibit a significant trade-off between forgetting and utility.

Base Model	Metric	Method						
		No-unlearn	*GDiff-0.9*	*GDiff-0.5*	*GDiff-0.1*	*NPO*	*LossNorm*	*NGDiff*
Llama2-7B	Verbmem ↓	0.561	0.555	**0.043**	**0.004**	**0.000**	0.388	**0.036**
	Knowmem ↓	0.755	0.717	**0.287**	**0.000**	**0.000**	**0.514**	**0.455**
	Utility ↑	0.646	**0.641**	0.275	0.000	0.000	**0.506**	**0.556**
Mistral-7B	Verbmem ↓	0.578	0.177	**0.000**	**0.000**	0.113	0.196	**0.098**
	Knowmem ↓	0.416	**0.257**	**0.000**	**0.000**	0.343	0.293	**0.165**
	Utility ↑	0.411	**0.339**	0.000	0.000	**0.316**	**0.343**	**0.354**

Table 3: Influence of AutoLR with different unlearning methods on the Phi-1.5 model. AutoLR improves the *TruthRatio* and reduces *Verbmem* across all methods. W/ or w/o AutoLR, NGDiff outperforms other baselines.

Method	**TOFU** (without → with AutoLR)		
	Verbmem ↓	*Utility* ↑	*TruthRatio* ↑
No-unlearn	1.00	1.00	0.39
GDiff c=0.9	0.81 → 0.20	0.99 → 0.42	0.21 → 0.31
GDiff c=0.5	0.03 → 0.00	0.31 → 0.03	0.22 → 0.30
GDiff c=0.1	0.00 → 0.00	0.00 → 0.00	0.22 → 0.23
NPO	0.00 → 0.00	0.00 → 0.00	0.18 → 0.22
LossNorm	0.43 → 0.23	0.75 → 0.73	0.21 → 0.34
NGDiff	0.02 → 0.01	0.61 → 0.75	0.29 → 0.35

the *Utility* and *TruthRatio*, such as GDiff with $c \leq 0.5$, NPO; (2) or cannot unlearn F on Phi-1.5 and Mistral-7B, such as LossNorm and GDiff with $c = 0.9$. We highlight that the effectiveness of these unlearning methods are highly model-dependent and dataset-dependent, unlike NGDiff.

For the *TOFU* dataset, we observe that some unlearning methods fail to unlearn the forget data effectively. For example, GDiff-0.9 and LossNorm do not unlearn effectively when applied to Phi-1.5, Llama2-7B and Mistral-7B. In fact, GDiff-0.9 has $80\% \sim 100\%$ *Verbmem* and LossNorm has $> 40\%$ *Verbmem* on Phi-1.5. However, they are effective on Falcon-1B and GPT2-XL, even though these models have similar sizes ($\approx$ 1B parameters) to Phi-1.5. On the other hand, some methods fail to preserve the model utility after unlearning. For example, GDiff-0.1 has close to 0 *Utility* on Phi-1.5, Falcon-1B, GPT2-XL and Llama2-7B; similarly, NPO also experiences a significant drop in *Utility* on Phi-1.5 model, Falcon-1B and GPT2-XL, but not so on Llama2-7B. In contrast, our NGDiff remains effective in unlearning F and maintaining R across the models. In addition, NGDiff achieves the best *TruthRatio* on all models except Llama2-7B (which is still on par with the best), indicating

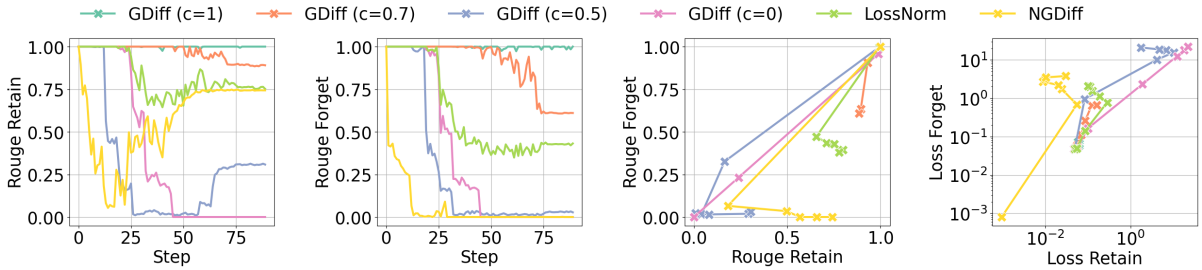

Figure 4: Comparison of unlearning methods on TOFU. The figures show the ROUGE scores and loss terms during unlearning process with different methods, which includes GDiff, LossNorm, and NGDiff. We observe that NGDiff effectively unlearns the forgetting data while maintaining the performance on the retaining data.

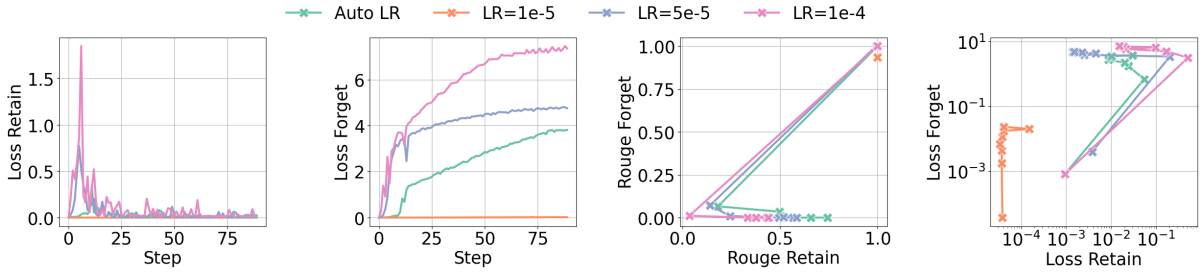

Figure 5: Comparison between AutoLR and different learning rates on NGDiff. The figures show the ROUGE scores and loss values during the unlearning process on TOFU dataset using Phi-1.5 model. We observe that AutoLR outperforms the static learning rates with better model utility and more stable convergence.

that the model's answers remain factually accurate for questions in the retaining data.

For the *MUSE-NEWS* dataset, NGDiff also outperforms the baseline methods on Llama2-7B and Mistral-7B models by achieving a lower *Verbmem* and a higher *Utility*. The *Knowmem* results indicate that NGDiff not only unlearns the verbatim copies of the forgetting texts, but also successfully removes the associated knowledge. While the model capacities of Phi-1.5 and Falcon-1B are smaller, limiting their ability to learn knowledge effectively after fine-tuning on the full dataset, as shown in Table 6, NGDiff still performs well.

To further illustrate the performance of our proposed method during the training, in addition to the last iterate results, we plot the ROUGE scores and loss terms during the unlearning process in Figure 4. We apply the extended GDiff, LossNorm, and NGDiff methods, to the Phi-1.5 model using the *TOFU* dataset. While GDiff with $c = 0.5$ and $c = 0.7$, and NGDiff are effective in unlearning, only NGDiff preserve the model utility above 75% ROUGE score. A closer look at the second and the fourth plots of Figure 4 shows that NGDiff exhibits the fastest and most stable convergence on F while maintaining a low retaining loss ≤ 0.1.

5.4 Ablation Study

Effectiveness of NGDiff. In our experiments, we utilize the automatic learning rate scheduler (AutoLR) for NGDiff method. To investigate the impact of NGDiff alone, we compare all methods with or without AutoLR in Table 3. With AutoLR or not (where we use manually tuned learning rates), NGDiff, GDiff ($c = 0.1$ or 0.5) and NPO can effectively unlearn in terms of *Verbmem*. However, among these four methods, NGDiff uniquely retains a reasonable *Utility* between $60 \sim 75\%$, while other methods retains only $0 \sim 30\%$ *Utility*. A similar pattern is observed in terms of *TruthRatio* as well. Overall, NGDiff significantly outperforms other baseline methods with or without AutoLR.

Impact of automatic learning rate. To evaluate the impact of AutoLR scheduler, we see in Table 3 all methods exhibit an increase in the *TruthRatio* metric and a decrease in *Verbmem*, though with some loss in the *Utility*. For instance, *LossNorm* benefits significantly from AutoLR with $\approx 20\%$ decrease in *Verbmem*, and NGDiff increases its retaining *Utility* and *TruthRatio* by $> 22\%$. We specifically demonstrate the impact of AutoLR on NGDiff in Figure 5. Without AutoLR, the model's performance is highly sensitive to the static learning rates: when $\eta = 10^{-5}$, the model fails to unlearn F as indicated by the low loss and high ROUGE score; in contrast, when $\eta = 10^{-4}$, there is a significant

drop in ROUGE score on the retain data, falling from 100% to around 50%. However, with the AutoLR scheduler, we observe a steady reduction in the *Verbmem* (with the ROUGE forget close to 0 at convergence) while maintaining high utility (the ROUGE retain is 0.747, which is 19.5% higher than the best results without AutoLR).

6 Conclusion and Discussion

We formulated the machine unlearning problem as a two-task optimization problem and proposed a novel unlearning method NGDiff based on normalized gradient difference and automatic learning rate adaption. By leveraging insights from multi-task optimization, NGDiff empirically improves forgetting quality while maintaining utility. We hope this paper helps establish a connection between LLM unlearning and multi-task optimization, and inspires further advancements in this field.

Limitations

Like other machine learning approaches in NLP, while our goal is to remove the influence of specific documents from LLMs, complete removal cannot always be guaranteed. Therefore, caution should be exercised when applying the proposed unlearning techniques in practical applications as unlearned LLMs can still potentially generate harmful or undesired outputs.

There are several technical alternatives that we did not explore in this paper due to its scope and limited resources. For example, other learning-rate-free methods could potentially be adapted as alternatives to the GeN approach used in this work. Additionally, other multi-task optimization methods could be applied to machine unlearning. However, scaling these approaches to the level of LLMs could be challenging, and are left as future work.

Finally, we mainly examined NGDiff's effectiveness on LLM unlearning in this paper with two benchmark datasets, TOFU and MUSE. To show its generalizability, we provide an additional example to apply the algorithm to computer vision tasks (see Appendix D). However, it would be desirable to test NGDiff on other modalities beyond NLP and CV applications.

Ethical Consideration

We acknowledge that our work is aligned with the *ACL Code of the Ethics* [8] and will not raise ethical concerns. We do not use sensitive datasets/models that may cause any potential issues/risks.

Acknowledgments

We would like to thank Shankar Ananthakrishnan, Fabian Triefenbach and Jianhua Lu from Amazon AGI Foundations team for providing feedback on this paper and supporting this work.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

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

Stephen Boyd and Lieven Vandenberghe. 2004. *Convex optimization*. Cambridge university press.

Zhiqi Bu and Shiyun Xu. 2024. Automatic gradient descent with generalized newton's method. *arXiv preprint arXiv:2407.02772*.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650.

Jiaao Chen and Diyi Yang. 2023. Unlearn what you want to forget: Efficient unlearning for LLMs. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12041–12052, Singapore. Association for Computational Linguistics.

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

Liang Chen, Shuming Ma, Dongdong Zhang, Furu Wei, and Baobao Chang. 2023. On the pareto front of multilingual neural machine translation. *ArXiv*, abs/2304.03216.

Shijie Chen, Yu Zhang, and Qiang Yang. 2021. Multi-task learning in natural language processing: An overview. *ACM Computing Surveys*.

[8] https://www.aclweb.org/portal/content/acl-code-ethics

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018a. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International conference on machine learning*, pages 794–803. PMLR.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018b. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. *Preprint*, arXiv:1711.02257.

Zhao Chen, Jiquan Ngiam, Yanping Huang, Thang Luong, Henrik Kretzschmar, Yuning Chai, and Dragomir Anguelov. 2020. Just pick a sign: Optimizing deep multitask models with gradient sign dropout. *Preprint*, arXiv:2010.06808.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6491–6506, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Aaron Defazio and Konstantin Mishchenko. 2023. Learning-rate-free learning by d-adaptation. *Preprint*, arXiv:2301.07733.

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B*, 39:1–38.

Jean-Antoine Désidéri. 2012. Multiple-gradient descent algorithm (mgda) for multiobjective optimization. *Comptes Rendus Mathematique*, 350(5-6):313–318.

Ronen Eldan and Mark Russinovich. 2023. Who's harry potter? approximate unlearning in llms.

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

Xudong Han, Timothy Baldwin, and Trevor Cohn. 2023. Fair enough: Standardizing evaluation and model selection for fairness research in NLP. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 297–312, Dubrovnik, Croatia. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep residual learning for image recognition. *Preprint*, arXiv:1512.03385.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *Preprint*, arXiv:2106.09685.

Maor Ivgi, Oliver Hinder, and Yair Carmon. 2023. Dog is sgd's best friend: A parameter-free dynamic step size schedule. *Preprint*, arXiv:2302.12022.

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 *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14389–14408, Toronto, Canada. Association for Computational Linguistics.

Adrián Javaloy and Isabel Valera. 2022. Rotograd: Gradient homogenization in multitask learning. *Preprint*, arXiv:2103.02631.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Ahmed Khaled, Konstantin Mishchenko, and Chi Jin. 2024. Dowg unleashed: An efficient universal parameter-free gradient descent method. *Preprint*, arXiv:2305.16284.

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

Yige Li, Hanxun Huang, Yunhan Zhao, Xingjun Ma, and Jun Sun. 2024. Backdoorllm: A comprehensive benchmark for backdoor attacks on large language models. *Preprint*, arXiv:2408.12798.

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

Yucheng Li, Frank Geurin, and Chenghua Lin. 2023b. Avoiding data contamination in language model evaluation: Dynamic test construction with latest materials. *arXiv preprint arXiv:2312.12343*.

Baijiong Lin, Feiyang Ye, Yu Zhang, and Ivor Tsang. 2021. Reasonable effectiveness of random weighting: A litmus test for multi-task learning. *Transactions on Machine Learning Research*.

Bo Liu, Qiang Liu, and Peter Stone. 2022. Continual learning and private unlearning. In *Conference on Lifelong Learning Agents*, pages 243–254. PMLR.

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. 2024a. Conflict-averse gradient descent for multi-task learning. *Preprint*, arXiv:2110.14048.

Liyang Liu, Yi Li, Zhanghui Kuang, J Xue, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. 2021. Towards impartial multi-task learning. In *iclr*.

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

Sijia Liu, Yuanshun Yao, Jinghan Jia, Stephen Casper, Nathalie Baracaldo, Peter Hase, Yuguang Yao, Chris Yuhao Liu, Xiaojun Xu, Hang Li, Kush R. Varshney, Mohit Bansal, Sanmi Koyejo, and Yang Liu. 2024c. Rethinking machine unlearning for large language models. *Preprint*, arXiv:2402.08787.

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*.

Konstantin Mishchenko and Aaron Defazio. 2024. Prodigy: An expeditiously adaptive parameter-free learner. *Preprint*, arXiv:2306.06101.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *Preprint*, arXiv:2203.02155.

Martin Pawelczyk, Seth Neel, and Himabindu Lakkaraju. 2024. In-context unlearning: Language models as few shot unlearners. In *ICML*.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only. *Preprint*, arXiv:2306.01116.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Preprint*, arXiv:2305.18290.

Rajhans Samdani, Ming-Wei Chang, and Dan Roth. 2012. Unified expectation maximization. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 688–698, Montréal, Canada. Association for Computational Linguistics.

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

Nianwen Si, Hao Zhang, Heyu Chang, Wenlin Zhang, Dan Qu, and Weiqiang Zhang. 2023. Knowledge unlearning for llms: Tasks, methods, and challenges. *Preprint*, arXiv:2311.15766.

The New York Times. 2023. Exhibit j.

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

Enayat Ullah, Tung Mai, Anup Rao, Ryan A Rossi, and Raman Arora. 2021. Machine unlearning via algorithmic stability. In *Conference on Learning Theory*, pages 4126–4142. PMLR.

Eric Wallace, Tony Zhao, Shi Feng, and Sameer Singh. 2021. Concealed data poisoning attacks on NLP models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 139–150, Online. Association for Computational Linguistics.

Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. 2020. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. *Preprint*, arXiv:2010.05874.

Xiaobao Wu, Liangming Pan, William Yang Wang, and Anh Tuan Luu. 2024. Akew: Assessing knowledge editing in the wild. *Preprint*, arXiv:2402.18909.

Derrick Xin, Behrooz Ghorbani, Justin Gilmer, Ankush Garg, and Orhan Firat. 2022. Do current multi-task optimization methods in deep learning even help? *Advances in neural information processing systems*, 35:13597–13609.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33:5824–5836.

Alan L Yuille and Anand Rangarajan. 2001. The concave-convex procedure (cccp). In *Advances in Neural Information Processing Systems*, volume 14. MIT Press.

Dawen Zhang, Pamela Finckenberg-Broman, Thong Hoang, Shidong Pan, Zhenchang Xing, Mark Staples, and Xiwei Xu. 2024a. Right to be forgotten in the era of large language models: Implications, challenges, and solutions. *Preprint*, arXiv:2307.03941.

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

Zhihan Zhang, Wenhao Yu, Mengxia Yu, Zhichun Guo, and Meng Jiang. 2023. A survey of multi-task learning in natural language processing: Regarding task relatedness and training methods. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 943–956, Dubrovnik, Croatia. Association for Computational Linguistics.

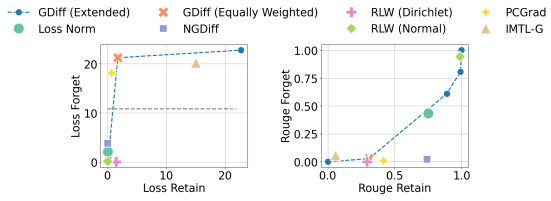

Figure 6: Loss values and ROUGE scores on the forgetting and retaining data from the *TOFU* dataset using different unlearning methods on the Phi-1.5 language model. We apply the *extended GDiff* with various coefficients (see (3), $0 \leq c \leq 1$) and connect the results with a blue dashed line. We denote MTO methods as different markers, and use a grey dashed line to represent the loss of random guess.

A Preliminary evidence

In our preliminary experiments, we observe two key issues preventing the standard methods from being practically applied. First, balancing retaining and forgetting losses is difficult. In Figure 6, we observe a trade-off between the performance on R and F, where some methods fail to unlearn F (points in the upper-right corner of the left figure), and some do not maintain utility in R (points in the bottom-left corner of the left figure). The blue dotted line in Figure 6 further illustrates the trade-off in GDiff by sweeping a hyper-parameter $c \in [0, 1]$, which is used to balance the losses on the forgetting and retaining data (see Eq. (3)). Picking an appropriate c to balance the two terms is often challenging. Secondly, the optimization methods for unlearning are usually sensitive to the learning rate. As illustrated in Figure 7, even for the same algorithm, various learning rates lead to substantial changes in the ROUGE scores and loss values, making the unlearning methods unstable and difficult to use in practice.

B Comparing NGDiff with GradNorm

Algorithm 2 NGDiff

for $t = 1, 2, \ldots$ **do**
  Compute retaining loss $L_{\mathrm{R}}(\boldsymbol{\theta}_t)$ by one forward pass
  Compute retaining gradient $\boldsymbol{g}_{\mathrm{R}}(\boldsymbol{\theta}_t) = \nabla_{\boldsymbol{\theta}} L_{\mathrm{R}}$
  Compute forgetting loss $L_{\mathrm{F}}(\boldsymbol{\theta}_t)$ by one forward pass
  Compute forgetting gradient $\boldsymbol{g}_{\mathrm{F}}(\boldsymbol{\theta}_t) = \nabla_{\boldsymbol{\theta}} L_{\mathrm{F}}$
  Construct unlearning gradient $\boldsymbol{g}_{\mathrm{NGDiff}} = \boldsymbol{g}_{\mathrm{R}}/||\boldsymbol{g}_{\mathrm{R}}|| - \boldsymbol{g}_{\mathrm{F}}/||\boldsymbol{g}_{\mathrm{F}}||$
  Update $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \boldsymbol{g}_{\mathrm{NGDiff}}$

We compare the GradNorm algorithm (Chen et al., 2018b) with our proposed method, NGDiff. We highlight some steps of GradNorm in red to

Algorithm 3 GradNorm for two-task

Initialize the scalaring coefficients $w_{\mathrm{R}}(\boldsymbol{\theta}_0) = 1$ and $w_{\mathrm{F}}(\boldsymbol{\theta}_0) = 1$
Pick value for $\alpha > 0$ and pick the weights $\boldsymbol{\theta}_{\mathrm{LS}}$ (the last shared layer of $\boldsymbol{\theta}_t$)
for $t = 1, 2, \ldots$ **do**
  Compute retaining loss $L_{\mathrm{R}}(\boldsymbol{\theta}_t)$ by one forward pass
  Compute retaining gradient $\boldsymbol{g}_{\mathrm{R}}(\boldsymbol{\theta}_{\mathrm{LS}}) = \nabla_{\boldsymbol{\theta}_{\mathrm{LS}}} L_{\mathrm{R}}$
  Compute forgetting loss $L_{\mathrm{F}}(\boldsymbol{\theta}_t)$ by one forward pass
  Compute forgetting gradient $\boldsymbol{g}_{\mathrm{F}}(\boldsymbol{\theta}_{\mathrm{LS}}) = \nabla_{\boldsymbol{\theta}_{\mathrm{LS}}} L_{\mathrm{F}}$
  Compute loss $L(\boldsymbol{\theta}_t) = w_{\mathrm{R}}(\boldsymbol{\theta}_t) L_{\mathrm{R}}(\boldsymbol{\theta}_t) + w_{\mathrm{F}}(\boldsymbol{\theta}_t) L_{\mathrm{F}}(\boldsymbol{\theta}_t)$
  Compute $\bar{\boldsymbol{g}}(\boldsymbol{\theta}_{\mathrm{LS}})$ by averaging $\boldsymbol{g}_{\mathrm{R}}$ and $\boldsymbol{g}_{\mathrm{F}}$
  Compute GradNorm loss

$$L_{GN}(\boldsymbol{\theta}_t) = |\boldsymbol{g}_{\mathrm{R}} - \bar{\boldsymbol{g}} \times [r_{\mathrm{R}}(t)]^{\alpha}|_1 + |\boldsymbol{g}_{\mathrm{F}} - \bar{\boldsymbol{g}} \times [r_{\mathrm{F}}(t)]^{\alpha}|_1$$

  Compute GradNorm gradients $\nabla_{w_{\mathrm{R}}} L_{GN}$ and $\nabla_{w_{\mathrm{F}}} L_{GN} \in \mathrm{R}$
  Compute the full gradient $\nabla_{\boldsymbol{\theta}_t} L$
  Update $w_{\mathrm{R}}(\boldsymbol{\theta}_t) \rightarrow w_{\mathrm{R}}(\boldsymbol{\theta}_{t+1})$ and $w_{\mathrm{F}}(\boldsymbol{\theta}_t) \rightarrow w_{\mathrm{F}}(\boldsymbol{\theta}_{t+1})$ using $\nabla_{w_{\mathrm{R}}} L_{GN}$ and $\nabla_{w_{\mathrm{F}}} L_{GN}$
  Update $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla_{\boldsymbol{\theta}_t} L$
  Renormalize $w_{\mathrm{R}}(\boldsymbol{\theta}_{t+1})$ and $w_{\mathrm{F}}(\boldsymbol{\theta}_{t+1})$ so that $w_{\mathrm{R}}(\boldsymbol{\theta}_{t+1}) + w_{\mathrm{F}}(\boldsymbol{\theta}_{t+1}) = 2$

indicate the differences than NGDiff:

- NGDiff sets the scalaring coefficient as $1/||\boldsymbol{g}_{\mathrm{R}}||$ and $1/||\boldsymbol{g}_{\mathrm{F}}||$, while GradNorm uses gradient descent to learn these coefficients as w_{R} and w_{F}.

- NGDiff is model-agnostic while GradNorm contains specific designs for multi-task architecture. In unlearning, there are **2 data splits** (i.e., F and R) and each data split defines one task. Hence all model parameters are shared. However, in the original form of GradNorm, there is **1 data split** on which multiple tasks are defined (can be more than 2). Hence the model parameters are partitioned into [shared layers, task 1 specific layers, task 2 specific layers].

- NGDiff computes the full per-task gradients whereas GradNorm only computes the last shared layer's gradients.

- NGDiff requires 2 back-propagation at each iteration but GradNorm requires 3 (2 for per-task gradients, 1 for $\nabla_{\boldsymbol{\theta}} L$), which may translate to more training time for large models.

- GradNorm introduces additional hyperparameters that can be difficult and costly to tune, and may cause instability of training if not

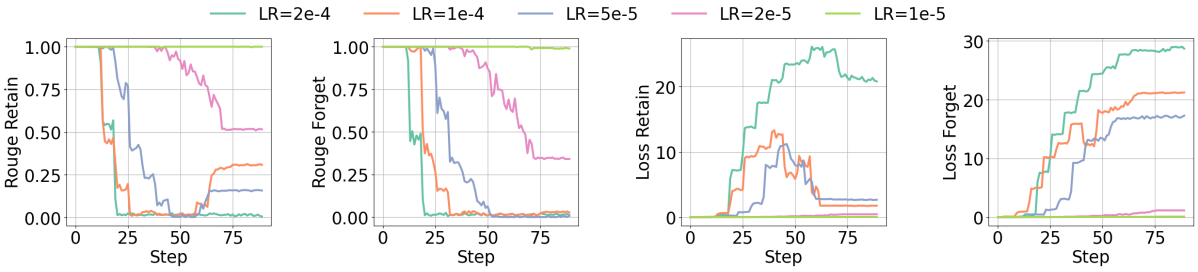

Figure 7: ROUGE scores and loss values during unlearning with vanilla GDiff (equally weighted), under different learning rates to which the unlearning performance is highly sensitive.

properly tuned. These hyperparameters include α and two learning rates to update w_{R} and w_{F} in Line 13 of Appendix B. In contrast, NGDiff is hyperparameter-free when equipped with GeN (AutoLR).

- NGDiff are theoretically supported by Theorem 5, while the choice of hyperparameters and the use of a heuristic $r_i(t)$ by GradNorm may require further justification. Here $r_i(t) = \widetilde{L}_i(\boldsymbol{\theta}_t)/\mathbb{E}_{\text{task}}[\widetilde{L}_i(\boldsymbol{\theta}_t)]$ is the "relative inverse training rate" of task i, where $\widetilde{L}_i(\boldsymbol{\theta}_t) = L_i(\boldsymbol{\theta}_t)/L_i(\boldsymbol{\theta}_0)$, $i \in \{\mathrm{F}, \mathrm{R}\}$.

In summary, NGDiff is remarkably simpler and more well-suited than GradNorm for unlearning, with stable performance and theoretical ground.

C Details related to GeN

C.1 Brief introduction of GeN

GeN (Bu and Xu, 2024) is a method that sets the learning rate for any given gradient $\boldsymbol{d}$ as

$$\eta_{\mathrm{GeN}} = \frac{\mathbf{G}^\top \boldsymbol{d}}{\boldsymbol{d}^\top \mathbf{H} \boldsymbol{d}}$$

where $\mathbf{G}$ is the gradient and $\mathbf{H}$ is the Hessian matrix of some loss L. One only needs to access the scalars $\mathbf{G}^\top \boldsymbol{d}$ and $\boldsymbol{d}^\top \mathbf{H} \boldsymbol{d}$, without computing the high-dimensional $\mathbf{G}$ and $\mathbf{H}$ (or Hessian-vector product). To do so, two additional forward passes are needed: given a constant (say $\xi = 0.001$), we compute $L(\boldsymbol{\theta} + \xi \boldsymbol{d})$ and $L(\boldsymbol{\theta} - \xi \boldsymbol{d})$. Then by curve fitting or finite difference as demonstrated below, we can estimate up to arbitrary precision controlled by ξ:

$$\mathbf{G}^\top \boldsymbol{d} \approx \frac{L(\boldsymbol{\theta} + \xi \boldsymbol{d}) - L(\boldsymbol{\theta} - \xi \boldsymbol{d})}{2\xi}$$

and

$$\boldsymbol{d}^\top \mathbf{H} \boldsymbol{d} \approx \frac{L(\boldsymbol{\theta} + \xi \boldsymbol{d}) - 2L(\boldsymbol{\theta}) + L(\boldsymbol{\theta} - \xi \boldsymbol{d})}{\xi^2}$$

Notice that the regular optimization requires 1 forward pass and 1 back-propagation; GeN requires in total 3 forward passes and 1 back-propagation. Given that back-propagation costs roughly twice the computation time than forward pass, the total time increases from 3 units of time to 5 units. Nevertheless, GeN needs not to be applied at each iteration: if we update the learning rate every 10 iterations as in Remark 4.2, the total time reduces to $3 + 2/10 = 3.2$ units, and the overhead is less than 10% compared to the regular optimization.

C.2 Adapting GeN to unlearning

Naively applying GeN to the unlearning will result in

$$\eta_{\mathrm{GeN}} = \frac{\mathbf{G}^\top \boldsymbol{g}_{\mathrm{UN}}}{\boldsymbol{g}_{\mathrm{UN}}^\top \mathbf{H} \boldsymbol{g}_{\mathrm{UN}}}$$

which minimizes the loss over all datapoints, in both F and R. This is against our goal to maximize the forgetting loss. We must consider the learning rate separately for F and R, as shown in Appendix G (Proof of Theorem 5). When both losses have a convex curvature in Figure 3, the optimal learning rate is only well-defined for L_{R} and we do not claim to maximize L_{F}. In other words, if we minimize L_{R}, we get to worsen L_{F} (though not maximally); if we choose to maximize L_{F}, we will use infinite learning rate that also maximizes L_{R}. Therefore, our learning rate in (7) only uses R instead of the whole dataset.

D Computer Vision Experiments

To demonstrate the effectiveness of unlearning across other modalities, we also evaluate our method on the image classification task. Specifically, we choose the CIFAR-10 and CIFAR-100 dataset (Krizhevsky et al., 2009) and train a ResNet-50 (He et al., 2015) model from scratch. For the CIFAR-10 dataset, we sample 500 images from the class *dog* as the forgetting data, and use images

Table 4: Results of *Forget Acc* and *Retain Acc* using different unlearning methods on the CIFAR-10 dataset. Compared to other baseline methods, *NGDiff* has the best performance on the model utility.

Method	CIFAR-10		CIFAR-100	
	Forget Acc ↓	*Retain Acc* ↑	*Forget Acc* ↓	*Retain Acc* ↑
No-unlearn	0.926	0.956	0.745	0.750
GDiff c=0.9	0.000	0.817	0.000	0.664
GDiff c=0.5	0.000	0.830	0.000	0.609
GDiff c=0.1	0.000	0.825	0.000	0.667
LossNorm	0.000	0.753	0.000	0.432
NGDiff	0.000	**0.931**	0.000	**0.701**

from the remaining 9 classes as the retaining data. For the CIFAR-100 dataset, we sample 500 images from the class *bed* as the forgetting data, and use images from the remaining 99 classes as the retaining data. After training, the initial forget data accuracy is 0.926, and the retain data accuracy is 0.956 on the CIFAR-10 dataset. The initial forget data accuracy is 0.745, and the retain data accuracy is 0.750 on the CIFAR-100 dataset. Then we apply different unlearning methods to the trained models. As shown in Table 4, all methods successfully reduce the forget accuracy to 0. However, the retaining accuracy of *NGDiff* remains the highest, which shows its effectiveness in preserving the model utility in image classification tasks.

E Other Related Works

Machine unlearning Machine unlearning is oftentimes viewed as a continual learning approach, that removes specific data points after a model has been trained to memorize them. Such removal is light-weighted in contrast to re-training, especially when the forgetting set is much smaller than the retaining. In addition to the methods already introduced in Section 3.2 (namely GA, GDiff and NPO), other methods include SISA (Bourtoule et al., 2021), influence functions (Ullah et al., 2021), differential privacy (Gupta et al., 2021) and so on. However, these methods could be difficult to scale on large models and large datasets due to the algorithmic complexity. To our best knowledge, this is the first work that formulate the unlearning problem as a two-task problem, which can be solved by a number of well-known MTO methods.

Multi-task optimization MTO is a paradigm where one model is trained to perform multiple tasks simultaneously, so as to significantly improve the efficiency in contrast to training multiple models, one for each task. The key challenge of MTO is the performance trade-off among tasks, where the multi-task model is worse than single-task model if trained on each task separately. Therefore, the core idea is to balance different tasks by modifying the per-task gradients, e.g. with normalization (LossNorm and NGDiff), PCGrad (Yu et al., 2020), RLW (Lin et al., 2021), IMTL (Liu et al., 2021), MGDA (Désidéri, 2012), CAGrad (Liu et al., 2024a), GradVaccine (Wang et al., 2020), GradDrop (Chen et al., 2020), RotoGrad (Javaloy and Valera, 2022), etc.

Learning-rate-free methods Parameter-free or learning-rate-free methods automatically set the learning rate scheduler without the hyperparameter tuning, which is computationally infeasible for LLMs, e.g. LLAMA2 pre-training uses 3 hyperparameters just for the learning rate: warmup steps, peak learning rate, and minimum learning rate. At high level, there are two approaches to learning-rate-free methods.

On one hand, GeN (Bu and Xu, 2024) leverages the Taylor expansion and convex-like landscape of deep learning, which is applicable for the general purpose, even if the gradient is modified like in the unlearning.

On the other hand, methods like D-adaptation (Defazio and Mishchenko, 2023), Prodigy (Mishchenko and Defazio, 2024), DoG (Ivgi et al., 2023), DoWG (Khaled et al., 2024) are based on the convex and G-Lipschitz conditions: $L(\bar{\theta}_T) - L(\boldsymbol{\theta}_*) \leq \frac{|\theta_0-\theta_*|^2}{2\eta T} + \frac{\eta G^2}{2}$ where θ_* is the unknown minimizer of L and $\bar{\theta}_T$ is an averaging scheme of $\{\theta_0, ..., \theta_T\}$. With the same theoretical foundation, these methods propose different ways to approximate the initial-to-final distance $|\theta_0 - \theta_*|$. There are two main issues to apply these methods on the unlearning. Firstly, the assumption of G-Lipschitz is hard to verify and the minimizer θ_* is not well-defined in multi-objective (see our discussion on Pareto optimality under Lemma 2). Secondly, the optimal learning rate $\frac{|\theta_0-\theta_*|}{G\sqrt{T}}$ is defined in a manner to minimize the

loss, whereas MTO methods operate on the gradient level. Hence MTO is incompatible to such parameter-free methods given that we cannot derive a corresponding loss (e.g. there exists no L_{NGDiff} such that $\frac{\partial L_{\text{NGDiff}}}{\partial \theta} = \boldsymbol{g}_{\text{NGDiff}}$).

F Experiments

F.1 Datasets

To evaluate the empirical performance of our proposed method, we experiment on the following datasets in Table 5.

- *Task of Fictitious Unlearning (TOFU)* (Maini et al., 2024). This dataset consists of question-answer pairs based on fictitious author biographies generated by GPT-4 (Achiam et al., 2023). Initially, predefined attributes, such as birthplace, gender, and writing genre, are assigned to 200 distinct authors. GPT-4 is then prompted to generate detailed information about each author. Following the synthesized data, 20 question-answer pairs are created for each fictitious author. The dataset is then divided into distinct datasets: the retaining set and the forgetting set. In our experiments, we use the *forget10* and *retain90* split, which excludes 10% of the original dataset.

- *MUSE-NEWS* (Shi et al., 2024). This dataset consists of BBC news articles (Li et al., 2023b) from August 2023. It includes seven subsets of news data: *raw*, *verbmem*, *knowmem*, *privleak*, *scal*, *sust*, and *train*. We utilize the *train* split to finetune a target model, and then the *raw* set, which includes both the forget and retain data, for the target model unlearning. Then, we use *verbmem*, *knowmem* split to evaluate the unlearned model's performance.

F.2 Evaluation Metrics

Following the existing work (Shi et al., 2024), we evaluate the unlearning performance based on the quality of outputs from the model after unlearning. We expect a good performance should satisfy the following requirements:

No verbatim memorization We evaluate this metric by prompting the model with the first l tokens of the news data in the forget set and compare the model's continuation outputs with the ground truth continuation. Specifically, for each input $x \in F$, we choose $x_{[:l]}$ as input, and compare the output $f(x_{[:l]})$ with the ground truth continuation $x_{[l+1:]}$ with the ROUGE-L recall score:

$$\begin{aligned} &\textit{Verbmem}(f, F) \\ &= \frac{1}{\|F\|} \Sigma_x \text{ ROUGE-L}(f(x_{[:l]}), x_{[l+1:]}) \end{aligned} \tag{9}$$

No knowledge memorization To evaluate this metric, we use the generated question-answer pair based on each example $x \in F$. We prompt the model with the question part q and compare the output answer $f(q)$ to the ground truth answer a using ROUGE-L recall scores:

$$\textit{Knowmem}(f, F) = \frac{1}{\|F\|} \Sigma_x \text{ ROUGE-L}(f(q), a) \tag{10}$$

Maintained model utility An effective unlearning method should also maintain the model's performance on the retain data. For the *MUSE-NEWS* dataset, we use the $\textit{Knowmem}_r$ split, which consists of the generated question-answer pairs based on the retain data. For the *TOFU* dataset, we prompt the model with the question from the retain set and compare the generated answer with the ground truth. We use ROUGE-L recall scores for evaluation:

$$\textit{Utility}(f, R) = \frac{1}{\|R\|} \Sigma_x \text{ ROUGE-L}(f(q), a) \tag{11}$$

Additionally, we evaluate the model using the *Retain10-perturbed* split from the *TOFU* dataset. It consists of five perturbed answers for each original answer, keeping original template but modifying the facts. We compute the Truth Ratio metric, which compares the likelihood of the model generating a correct answer versus an incorrect one for each question in the retain set. A higher Truth Ratio indicates better model utility that effectively remembers knowledge from the retain data.

F.3 Hyper-parameter Settings

To finetune a targeted model with the full dataset, we use the optimizer Adam with a learning rate of $\eta = \{10^{-5}, 2 * 10^{-5}\}$, a training batch size of {16, 32}, and train 25 epochs for all language models. For the unlearning process, we use the optimizer Adam with a learning rate $\eta = \{10^{-5}, 5 * 10^{-5}, 10^{-4}\}$, and train 15 epochs for all unlearning

Table 5: Statistics of the *TOFU* and *MUSE-NEWS* datasets. For the *TOFU* dataset, we use *Full* split for training the target model, *Forget10* and *Retain90* as the forgetting and retaining split for unlearning experiments. For the *MUSE-NEWS* dataset, we utilize *Train* split for training, *Raw* split for unlearning. For evaluation, we use Verbmem$_F$ and Knowmem$_F$ splits from forgetting data, and Knowmem$_R$ split from the retaining data.

Dataset	**TOFU**			**MUSE-NEWS**				
	Full	Forget10	Retain90	Train	Raw	Verbmem$_F$	Knowmem$_F$	Knowmem$_R$
# samples	4,000	400	3,600	7,110	2,669	100	100	100

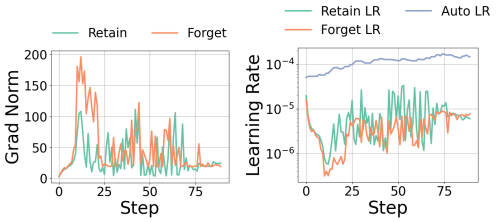

Figure 8: Gradient norms and learning rates during the unlearning on *TOFU* dataset using NGDiff and Phi-1.5 model. The AutoLR scheduler assigns a smaller learning rate to the forgetting gradient, effectively preserving model utility on the retaining set.

methods. For the Phi-1.5, Falcon-1B, and GPT2-XL models, we perform full-model parameter tuning. For the Llama2-7B and Mistral-7B models, we apply the LoRA training method (Hu et al., 2021) with $rank = 8$.

F.4 Other unlearning results

More results on *MUSE-NEWS* dataset with Phi-1.5 model and Falcon-1B model are in Table 6.

F.5 Visualization of learning rate scheduling

We monitor the gradient norms and the learning rate in Figure 8 for Algorithm 1. We observe that the automatic learning rate is indeed effective, picking up from $5e-5$ to around $2e-4$, and that NGDiff assigns a smaller learning rate to the forgetting gradient, not perturbing the model too much to maintain the high utility on the retaining set.

G Proofs

In this section, we provide the proofs of all the lemma and theorems in this paper.

Lemma 2 (restated from (Xin et al., 2022))**.** *For any $0 < c < 1$, the model $\boldsymbol{\theta}^*_{LSP}(c) = argmin_\theta \mathrm{LSP}(\boldsymbol{\theta}; c)$ is Pareto optimal.*

Proof of Lemma 2. We show that the solution of LSP cannot be a dominated point, and therefore it must be Pareto optimal. Consider a solution $\boldsymbol{\theta}^* = \mathrm{argmin}_\theta \mathrm{LSP}(\boldsymbol{\theta}; c)$, and suppose it is dominated by some $\boldsymbol{\theta}'$, i.e. $L_\mathrm{F}(\boldsymbol{\theta}^*) \le L_\mathrm{F}(\boldsymbol{\theta}'), L_\mathrm{R}(\boldsymbol{\theta}^*) \ge L_\mathrm{R}(\boldsymbol{\theta}')$ with at least one inequality being strict. This contradicts that $\boldsymbol{\theta}^*$ is minimal as $cL_\mathrm{R}(\boldsymbol{\theta}^*) - (1-c)L_\mathrm{F}(\boldsymbol{\theta}^*) > cL_\mathrm{R}(\boldsymbol{\theta}') - (1-c)L_\mathrm{F}(\boldsymbol{\theta}')$. □

Theorem 3. *For any $\{c_t\}$ with $0 \le c_t \le 1$ that converges as $t \to \infty$, the model $\boldsymbol{\theta}^*(\{c_t\}) = \lim_{t\to\infty} \boldsymbol{\theta}_t$ in (4) is Pareto optimal.*

Proof of Theorem 3. Let $c = \lim_t c_t$, then Eq. (4) gives that $\boldsymbol{g}_\mathrm{UN}(\boldsymbol{\theta}_t) = c_t \boldsymbol{g}_\mathrm{R}(\boldsymbol{\theta}_t) - (1-c_t)\boldsymbol{g}_\mathrm{F}(\boldsymbol{\theta}_t) \to c\boldsymbol{g}_\mathrm{R}(\boldsymbol{\theta}^*) - (1-c)\boldsymbol{g}_\mathrm{F}(\boldsymbol{\theta}^*) = \mathbf{0}$ as $t \to \infty$. Note $\boldsymbol{\theta}^*(\{c_t\})$ is equivalent to the LSP solution $\boldsymbol{\theta}^*_\mathrm{LSP}(c) = \mathrm{argmin}_\theta \mathrm{LSP}(\boldsymbol{\theta}; c)$ as the latter has the same stationary condition, which is Pareto optimal by Lemma 2. □

Lemma 4. *$\boldsymbol{g}_\mathrm{NGDiff}(\boldsymbol{g}_\mathrm{R}, \boldsymbol{g}_\mathrm{F})$ satisfies Eq. (6) for any $\boldsymbol{g}_\mathrm{R} \in \mathbb{R}^d$ and $\boldsymbol{g}_\mathrm{F} \in \mathbb{R}^d$.*

Proof of Lemma 4. We firstly show $\boldsymbol{g}_\mathrm{R}^\top \boldsymbol{g}_\mathrm{UN} \ge 0$ for $\boldsymbol{g}_\mathrm{UN} = \boldsymbol{g}_\mathrm{NGDiff}$. We write

$$\begin{aligned}\boldsymbol{g}_\mathrm{R}^\top \boldsymbol{g}_\mathrm{NGDiff} &= \boldsymbol{g}_\mathrm{R}^\top \left(\frac{\boldsymbol{g}_\mathrm{R}}{\|\boldsymbol{g}_\mathrm{R}\|} - \frac{\boldsymbol{g}_\mathrm{F}}{\|\boldsymbol{g}_\mathrm{F}\|} \right) \\ &= \|\boldsymbol{g}_\mathrm{R}\| - \frac{\boldsymbol{g}_\mathrm{R}^\top \boldsymbol{g}_\mathrm{F}}{\|\boldsymbol{g}_\mathrm{F}\|} \ge \|\boldsymbol{g}_\mathrm{R}\| - \frac{\|\boldsymbol{g}_\mathrm{R}\|\|\boldsymbol{g}_\mathrm{F}\|}{\|\boldsymbol{g}_\mathrm{F}\|} \\ &= 0\end{aligned}$$

where the inequality is the Cauchy-Schwarz inequality. Similarly, $\boldsymbol{g}_\mathrm{F}^\top \boldsymbol{g}_\mathrm{UN} \le 0$ easily follows. □

Theorem 5. *Consider $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \boldsymbol{g}_\mathrm{NGDiff}$.*

(1) Unless $\boldsymbol{g}_\mathrm{R}$ is exactly parallel to $\boldsymbol{g}_\mathrm{F}$, for any sufficiently small learning rate η, there exist two constants $\epsilon_{\mathrm{R},1} = o(\eta), \epsilon_{\mathrm{F},1} = o(\eta)$ such that

$$L_\mathrm{R}(\boldsymbol{\theta}_{t+1}) - L_\mathrm{R}(\boldsymbol{\theta}_t) < \epsilon_{\mathrm{R},1};$$

$$L_\mathrm{F}(\boldsymbol{\theta}_{t+1}) - L_\mathrm{F}(\boldsymbol{\theta}_t) > \epsilon_{\mathrm{F},1}.$$

(2) If additionally $\boldsymbol{g}_\mathrm{NGDiff}^\top \mathbf{H}_\mathrm{R} \boldsymbol{g}_\mathrm{NGDiff} > 0$ and $\boldsymbol{g}_\mathrm{NGDiff}^\top \mathbf{H}_\mathrm{F} \boldsymbol{g}_\mathrm{NGDiff} > 0$, then for any learning rate $0 < \eta < \frac{2\boldsymbol{g}_\mathrm{R}^\top \boldsymbol{g}_\mathrm{NGDiff}}{\boldsymbol{g}_\mathrm{NGDiff}^\top \mathbf{H}_\mathrm{R} \boldsymbol{g}_\mathrm{NGDiff}}$, there exist two constants $\epsilon_{\mathrm{R},2} = o(\eta^2), \epsilon_{\mathrm{F},2} = o(\eta^2)$ such that

$$L_\mathrm{R}(\boldsymbol{\theta}_{t+1}) - L_\mathrm{R}(\boldsymbol{\theta}_t) < \epsilon_{\mathrm{R},2};$$

Table 6: Results of *Verbatim memorization*, *Model utility*, and *TruthRatio* on *MUSE-NEWS* dataset with different unlearning methods on Phi-1.5, and Falcon-1B models. Lower *Verbmem* along with higher *Utility* and *TruthRatio* indicate a more superior unlearning strategy.

Base Model	Metric	Method						
		No-unlearn	*GDiff-0.9*	*GDiff-0.5*	*GDiff-0.1*	*NPO*	*LossNorm*	*NGDiff*
Phi-1.5	Verbmem ↓	0.018	0.000	0.012	0.000	0.000	0.012	**0.004**
	Utility ↑	0.372	0.277	0.061	0.000	0.000	0.061	**0.001**
	Knowmem ↓	0.030	0.000	0.002	0.000	0.000	0.002	**0.023**
Falcon-1B	Verbmem ↓	0.204	0.132	0.000	0.000	0.000	0.126	**0.000**
	Utility ↑	0.386	0.214	0.000	0.000	0.000	0.142	**0.025**
	Knowmem ↓	0.232	0.078	0.000	0.000	0.000	0.130	**0.087**

$$L_{\mathrm{F}}(\boldsymbol{\theta}_{t+1}) - L_{\mathrm{F}}(\boldsymbol{\theta}_t) > \epsilon_{\mathrm{F},2}.$$

Proof of Theorem 5. Applying (5) with $\boldsymbol{g}_{\mathrm{NGDiff}}$ gives

$$\begin{aligned} &L_{\mathrm{R}}(\boldsymbol{\theta}_{t+1}) - L_{\mathrm{R}}(\boldsymbol{\theta}_t) \\ &= -\eta \boldsymbol{g}_{\mathrm{R}}^{\top} \boldsymbol{g}_{\mathrm{NGDiff}} + \frac{\eta^2}{2} \boldsymbol{g}_{\mathrm{NGDiff}}^{\top} \mathbf{H}_{\mathrm{R}} \boldsymbol{g}_{\mathrm{NGDiff}} + o(\eta^2) \end{aligned} \tag{12}$$

For part (1), note that Lemma 4 gives $\boldsymbol{g}_{\mathrm{R}}^{\top} \boldsymbol{g}_{\mathrm{NGDiff}} > 0$ unless $\boldsymbol{g}_{\mathrm{F}} \parallel \boldsymbol{g}_{\mathrm{R}}$. Hence for any $\eta > 0$, we have

$$L_{\mathrm{R}}(\boldsymbol{\theta}_{t+1}) - L_{\mathrm{R}}(\boldsymbol{\theta}_t) = -\eta \boldsymbol{g}_{\mathrm{R}}^{\top} \boldsymbol{g}_{\mathrm{NGDiff}} + o(\eta) < o(\eta)$$

and similarly for L_{F}.

For part (2), now that $\boldsymbol{g}_{\mathrm{NGDiff}}^{\top} \mathbf{H}_{\mathrm{R}} \boldsymbol{g}_{\mathrm{NGDiff}} > 0$, we have

$$\begin{aligned} &-\eta \boldsymbol{g}_{\mathrm{R}}^{\top} \boldsymbol{g}_{\mathrm{NGDiff}} + \frac{\eta^2}{2} \boldsymbol{g}_{\mathrm{NGDiff}}^{\top} \mathbf{H}_{\mathrm{R}} \boldsymbol{g}_{\mathrm{NGDiff}} < 0 \\ &\iff 0 < \eta < \frac{2 \boldsymbol{g}_{\mathrm{R}}^{\top} \boldsymbol{g}_{\mathrm{NGDiff}}}{\boldsymbol{g}_{\mathrm{NGDiff}}^{\top} \mathbf{H}_{\mathrm{R}} \boldsymbol{g}_{\mathrm{NGDiff}}} \end{aligned}$$

and similarly

$$-\eta \boldsymbol{g}_{\mathrm{F}}^{\top} \boldsymbol{g}_{\mathrm{NGDiff}} + \frac{\eta^2}{2} \boldsymbol{g}_{\mathrm{NGDiff}}^{\top} \mathbf{H}_{\mathrm{F}} \boldsymbol{g}_{\mathrm{NGDiff}} > 0 \impliedby 0 < \eta$$

We complete the proof by substituting the inequalities into (12). □

Remark G.1. *There is a computational overhead to use GeN, as it requires additional forward passes to estimate η_t^*. Nevertheless, we only update the learning rate every 10 iterations so that the overhead is amortized and thus negligible.*

Proof of Remark G.1. We extend our original complexity analysis in Remark 4.2 and provide quantitative analysis of computation overheads in FLOPs (floating point operations). Specifically, assume it takes around N FLOPS to perform one forward pass on one example and 2N FLOPS to back-propagate. The basic GDiff requires roughly 6BTN FLOPS to run with batch size B and total number of iterations T, because it needs 1 forward and 1 backward for the retain set and another for the forget set. Our learning-rate-free NGDiff requires about (6BTN+4BTN/10), hence a 6.6% increase to unlearn the data. We assure that the extra forward passes do not add memory burden, because they are in gradient-free mode (e.g. under `torch.no_grad()` mode). All in all, NGDiff is almost as efficient as GDiff and other unlearning methods, as the target unlearning corpus is usually relatively small. This has been empirically observed by our large-scale (up to 7B) model unlearning experiments. □