Towards Source-Free Machine Unlearning

Sk Miraj Ahmed[1,2,*,†] Umit Yigit Basaran[2,*] Dripta S. Raychaudhuri[3,‡] Arindam Dutta[2]
Rohit Kundu[2], Fahim Faisal Niloy[2], Başak Güler[2], Amit K. Roy-Chowdhury[2]
[1]Brookhaven National Laboratory [2]University of California, Riverside [3]AWS AI
{sahme047@, ubasa001@, drayc001@, adutt020@}ucr.edu
{rkund006@, fnilo001@, bguler@ece., amitrc@ece.}ucr.edu

Abstract

As machine learning becomes more pervasive and data privacy regulations evolve, the ability to remove private or copyrighted information from trained models is becoming an increasingly critical requirement. Existing unlearning methods often rely on the assumption of having access to the entire training dataset during the forgetting process. However, this assumption may not hold true in practical scenarios where the original training data may not be accessible, i.e., the source-free setting. To address this challenge, we focus on the source-free unlearning scenario, where an unlearning algorithm must be capable of removing specific data from a trained model without requiring access to the original training dataset. Building on recent work, we present a method that can estimate the Hessian of the unknown remaining training data, a crucial component required for efficient unlearning. Leveraging this estimation technique, our method enables efficient zero-shot unlearning while providing robust theoretical guarantees on the unlearning performance, while maintaining performance on the remaining data. Extensive experiments over a wide range of datasets verify the efficacy of our method.

1. Introduction

Machine learning models have achieved significant success by training on large amounts of annotated data, much of which may include sensitive or private information [15]. With the introduction of data protection rules such as the General Data Protection Regulation (GDPR) [27], there is a growing need for algorithms that can delete (or forget) information learned from such sensitive datasets. Furthermore, privacy concerns may prompt individuals to request the removal of their data from the training set, invoking their "right to be forgotten" [24]. A straightforward solution to this issue would be to retrain the model from scratch using only the non-private subset of the original dataset. However, retraining is computationally inefficient and impractical (and impossible in the source-free setting we introduce in this paper). This highlights the necessity for efficient *Machine Unlearning* (MU) [12, 33] algorithms that enable modifications to the trained model parameters to forget specified data while maintaining performance on the remaining data.

Although several recent machine unlearning algorithms have demonstrated reasonable success on existing benchmarks [20, 26], nearly all current approaches [12, 20, 26, 33] assume the availability of the remaining data, either in full or in part. In practical settings, storing such large volumes of data is challenging due to storage costs and privacy issues. Consequently, these methods fail to address scenarios where the *model owner no longer has access to the original training data*. In these situations, ensuring accurate and efficient unlearning becomes markedly more difficult. Without the original data, it is challenging to verify that the specified information has been entirely removed from the model and that the model's performance on the remaining data remains unaffected. This limitation underscores the pressing need to develop robust unlearning techniques that can function effectively even when the original training dataset is inaccessible, i.e., the source-free setting.

A recent study has introduced a solution for this challenge, referring to the setting as "zero-shot machine unlearning" [5], which works by solely requiring access to the trained model weights and the data to be forgotten, without needing the original training dataset. However, a significant limitation of this technique is its inability to forget random instances encompassing diverse classes; it can only selectively forget particular data classes. This constraint could hinder its practicality in scenarios where users only want certain instances unlearned, as this method discards all the data pertaining to a user, rather than selectively removing certain examples. To tackle this limitation, another

[*]Equal contribution.
[†]Work partially done while at University of California, Riverside.
[‡]Work done outside AWS

recent investigation [3] attempts zero-shot unlearning at the instance level instead of the class level. This approach enables the removal of requested data without requiring access to the complete training dataset. However, it suffers from scalability issues, as increasing the number of instances results in a significant drop in performance on both test data and the remaining dataset, which is undesirable for effective and reliable machine unlearning. Additionally, these methods fall short in providing robust theoretical assurances regarding their respective performance.

Considering the aforementioned issues, *we propose an unlearning algorithm* that excels in such scenarios, *where the original training data is unavailable, while providing robust theoretical guarantees*. We term this as "source-free machine unlearning", in analogy to source-free domain adaptation methods [23]. (We believe this is a better term than zero-shot unlearning.) Inspired by [15], we study the unlearning mechanisms of ℓ_2 regularized linear models with differentiable convex loss functions. Specifically, Guo et al. [15] define a Newton update step on the model parameters, which can be used to perform unlearning. This step is proven to be optimal for the quadratic loss function, and for strongly convex Lipschitz loss functions, the discrepancy between this step and optimal forgetting is bounded. Crucially, this Newton step requires the Hessian of the remaining data with respect to the trained model parameters. However, in our problem setup, we do not have access to the remaining data. Thus, we cannot compute this Hessian directly.

To address this challenge, we introduce an algorithm that approximately estimates the Hessian of the retained data (referred to as the "retain Hessian") using only the data designated for removal and the trained model by assuming the loss differences at perturbed points for remaining and forget data are close to each other. By focusing on closely aligning the Hessian estimate with the true value, our approach offers robust theoretical guarantees. This aspect is crucial as it reinforces confidence in the machine unlearning process for data removal, ensuring both accuracy and reliability.

Our main contributions in this work can be summarized as follows:

- To the best of our knowledge, this work proposes the first zero-shot unlearning method for linear classifiers that can effectively forget arbitrary instances of data across all classes while providing strong theoretical guarantees for data removal and privacy.
- Recognizing that the Hessian cannot be directly computed from the remaining data, we introduce a novel method for estimating the Hessian based on the data marked for removal. This approach ensures an approximate (i.e., bounded-error) unlearning mechanism applicable to any convex loss functions. While our algorithm is specifically designed for linear classifiers, we demonstrate its applicability in mixed linear scenarios, such as by *linearizing a deep neural network*.
- We establish *theoretical guarantees* for our unlearning framework through comprehensive proofs and validate our approach with experiments and ablation studies using multiple benchmark datasets.

2. Related Works

Machine unlearning. Machine unlearning, introduced in [2], aims to efficiently remove the influence of certain training instances from a model's parameters. Unlearning approaches in the literature can be primarily categorized into exact and approximate unlearning methods. Exact unlearning methods aim to ensure that the data is completely unlearned from the model, akin to retraining from scratch. Recent approaches include [1, 7], which split the data into multiple shards and train separate models on different non-overlapping combinations of these shards. However, it comes with substantial storage costs since multiple models must be maintained. In contrast, approximate unlearning methods estimate the influence of the unlearning instances and remove it through direct parameter updates. Some approximate methods focus on improving efficiency [30] or preserving performance [29], but they lack formal guarantees on data removal. A second group of approximate approaches [12, 13, 15, 25] provide theoretical guarantees on the statistical indistinguishability of unlearned and retrained models based on ideas similar to differential privacy [8]. All these methods require access to access to all, or a subset of, the training data. This assumption may not hold true in many practical settings; nevertheless, data privacy concerns may need to be addressed [11]. Recently, machine unlearning has attracted attention and achieved notable success in various applications, such as mitigating bias [4], erasing unwanted or copyrighted content [10, 21], and preventing malicious attacks [31] in recent large-scale generative models.

Source-free unlearning. A recent paper has proposed a method for unlearning which works by solely requiring access to the trained model weights and the data to be forgotten, without needing the original training dataset, referring to it as "zero-shot machine unlearning" [5]. They propose two approaches: error minimizing-maximizing noise and gated knowledge transfer.

The first approach learns a set of noise matrices which maximize the error for the forget set, and a separate set of noise matrices which minimize the error as a proxy for the remaining data. The second approach uses knowledge distillation of the original model into a new model, gated by a filter that prevents the forget set knowledge from being passed, and additionally, supplemented by a generator network for sample generation. A major limitation of these methods is their inability to forget specific instances of different classes; rather, they forget all the data pertaining to a class. However, such fine-grained forgetting scenarios are likely to occur in real-world applications, where the need for selective data removal or modification is prevalent. A recent work [3] proposes an adversarial sample generation strategy to extend zero-shot unlearning to the instance-wise case. However, this method struggles to scale beyond forgetting a few samples without significantly degrading model performance. Another recent work [9] proposes a method called Just-in-Time (JiT) unlearning where they minimize the gradient effect by creating perturbed samples around each forget sample and fine-tune the model with those samples for reducing its reliance on that sample.

Critically, all existing zero-shot machine unlearning methods fail to provide any formal guarantees regarding the completeness or effectiveness of the forgetting process. In practical applications, where data privacy and compliance are paramount, such guarantees are essential to ensure that sensitive information is reliably removed from the model without compromising its overall performance. Additionally, a core contribution of our work is to provide such guarantees when the source data is no longer available.

3. Preliminaries

The mathematical concept central to the ideal machine unlearning setting is known as *parameter indistinguishability* [15]. In this section, we provide a brief overview of this definition. Additionally, we present the preliminaries of the unlearning mechanism for linear classifiers under convex losses.

3.1. Parameter Indistinguishability

Consider a data distribution $\mathcal{D} \sim \{x_i, y_i\}_{i=1}^n$ representing a training set used to train a model with a randomized algorithm $\mathcal{A}$ resulting in the output hypothesis space $\mathcal{H}$. Suppose there is a desire to eliminate the influence of x_i from $\mathcal{H}$ using an unlearning mechanism Ξ. The unlearning mechanism is said to achieve *parameter indistinguishability*, if Ξ functions in a manner such that the outputs of $\Xi(\mathcal{A}(\mathcal{D}), \mathcal{D}, x_i)$ and $\mathcal{A}(\mathcal{D}\backslash x_i)$ are very close to each other. The current trend in unlearning research emphasizes demonstrating the efficacy of designed mechanisms using this metric. In simpler terms, the unlearned model should closely mimic, in terms of output space, the model that has been retrained from scratch without the specific data. Further discussion on this aspect will be provided in detail in the experimental section. In our case, we explore linear classifiers with the randomized algorithm $\mathcal{A}$ being the supervised learning, using standard convex loss functions.

3.2. Unlearning of Linear Classifier

The empirical loss with respect to a linear classifier $w \in \mathbb{R}^d$ and a convex loss function $l : \mathbb{R}^d \to \mathbb{R}$ can be written as $\mathcal{L}(w) = \sum_{i=1}^n l(y_i, w^\top x_i) + \frac{\lambda n}{2}\|w\|_2^2$. Let $w^\star = \underset{w}{\arg\min}\ \mathcal{L}(w)$ be the optimal linear classifier trained on the distribution $\mathcal{D}$. To forget a subset of the training data $\mathcal{D}_f \subset \mathcal{D}$, the naive approach involves retraining the classifier over the distribution $\mathcal{D}_r = \mathcal{D}\backslash\mathcal{D}_f$. However, this approach is impractical and time-consuming. A more widely used alternative is to mitigate the influence of the forget dataset using the influence function [15, 28] on the optimal model parameters. Mathematically, this unlearning mechanism can be expressed as:

$$\Xi(w^\star, \mathcal{D}, \mathcal{D}_f) = w_{uf} = w^\star + \mathrm{H}_r^{-1}\nabla_f + \sigma^2\varepsilon \tag{1}$$

Here, w_{uf} represents the model parameters obtained by unlearning the forget dataset, $w^\star$ is the optimal model parameter obtained using the entire training data, H_r is the Hessian of the remaining dataset, and ∇_f is the gradient of the forget dataset at the optimal point $w^\star$, σ is the variance of the noise term and ε is the noise sampled from a standard Gaussian distribution. The term $-\mathrm{H}_r^{-1}\nabla_f$ corresponds to the influence of the forget dataset on the model parameters. Also, the noise term seeks to remove any information that could potentially leak due to slight inconsistencies. This unlearning methods is theoretically grounded and the residual norm of the gradient of the unlearned model on the remaining training set $\mathcal{D}_r$ can be tightly upper bounded. However, the assumption of having access to $\mathcal{D}$ during unlearning is strong and we relax this problem where we just have access to $\mathcal{D}_f$. However, without $\mathcal{D}_r$, computing the Hessian H_r as in Eq. (1) is non-trivial. To solve this, we devise a method where we can approximate this H_r using only $w^\star$ and $\mathcal{D}_f$, which is elaborated in the next section in detail.

3.3. Mixed Linear Unlearning

Due to the highly non-convex structure of the loss landscapes of neural networks, unlearning is a challenging task. As a solution to this problem in [13] authors proposed Mixed Linear Unlearning where the loss function is stated as the following:

$$\mathcal{L}(w) = \sum_{i=1}^{n} \|f_{w_c^*}(x_i) + \nabla_w f_{w_c^*}(x_i)\cdot w - y_i\|_2^2 + \frac{\lambda n}{2}\|w\|_2^2$$

where w_c^* is the model parameters trained on a public dataset, $f_{w_c^*}$ is the function parameterized with w_c^* i.e. neural network, and $\nabla_w f_{w_c^*}$ is the Jacobian of the function $f_{w_c^*}$. The model parameters w are trained on the user-specific dataset where there can be an unlearning request after training. The Neural Tangent Kernel (NTK) [17, 22] approach in mixed linear unlearning leverages a first-order Taylor expansion to approximate the network's behavior in a linearized form, which simplifies the complex task of forgetting in non-linear networks. This approximation transforms the problem into a convex optimization task, allowing data removal to be treated with a closed-form solution rather than requiring full re-optimization. By linearizing the network's behavior around specific weights, NTKs make the process of forgetting efficient, enabling the model to quickly process data deletion requests without the computational burden of extensive retraining and while preserving most of the model's predictive performance. Again, the unlearning mechanism proposed for this method is the same as Eq. (1). In our experiments, to show our methods performance on neural networks we utilized the mixed linear unlearning approach. Our proposed method in the next section is easily applicable to this setup.

4. Methodology

Given a differentiable convex loss $\mathcal{L}$, we can write the Taylor approximation of this around the optimal classifier $w^\star$ as follows:

$$\begin{aligned}\mathcal{L}(w) \approx \mathcal{L}(w^\star) &+ \nabla(w^\star)^\top(w - w^\star)\\ &+ \frac{1}{2}(w - w^\star)^\top \mathrm{H}(w^\star)(w - w^\star)\end{aligned}$$

where the higher-order terms of the Taylor expansion are neglected here due to its relatively small magnitude. We define the loss difference $\delta\mathcal{L}$ as follows:

$$\delta\mathcal{L} = \mathcal{L}(w) - \mathcal{L}(w^\star)$$

Assuming this $\delta\mathcal{L}$ is computed over the whole training data, we denote the loss difference with respect to the retain data $x^r \in \mathcal{D}_r$ as $\delta\mathcal{L}_r$. So,

$$\begin{aligned}\delta\mathcal{L}_r \approx \nabla_r(w^\star)^\top(w - w^\star)&\\ + \frac{1}{2}(w - w^\star)^\top \mathrm{H}_r(w^\star)(w - w^\star)&\end{aligned} \tag{2}$$

Assuming that the training converges to the global optima $w^\star$, we can safely assume that $\nabla(w^\star) = 0$, which also means $\nabla_r(w^\star) + \nabla_f(w^\star) = 0 \implies \nabla_r(w^\star) = -\nabla_f(w^\star)$. Plugging this in Eq. (2) we get the following:

$$\frac{1}{2}(\delta w)^\top \mathrm{H}_r(w^\star)(\delta w) - \nabla_f(w^\star)^\top \delta w - \delta\mathcal{L}_r \approx 0$$

where, $\delta w = (w - w^\star)$. With this observation we generate some (m points) small perturbations around the optima $w_i = w^\star + (\delta w)_i$ and calculate the average to formulate the following objective function of the Hessian as follows:

$$\Psi(\mathrm{H}_r) = \frac{1}{m}\sum_{i=1}^{m}\left(\mathrm{f}_i(\mathrm{H}_r)\right)^2$$

where $\mathrm{f}_i(\mathrm{H}_r) = \frac{1}{2}(\delta w)_i^\top \mathrm{H}_r(w^\star)(\delta w)_i - \nabla_f(w^\star)^\top(\delta w)_i - \delta\mathcal{L}_r(w_i)$. Since $\Psi(\mathrm{H}_r) \to 0$ at the optima, minimizing it should output the desired Hessian H_r w.r.t to the retain data. However, since we do not have access to $\mathcal{D}_r$, we can not explicitly compute $\delta\mathcal{L}_r(w_i)$. Instead, we can replace it with next best value which is $\delta\mathcal{L}_f(w_i)$. This is reasonable replacement since $\delta\mathcal{L}(w_i) \leq L\|w_i - w^\star\|$ where L is the Lipschitz constant corresponding to the loss. As a result both $\delta\mathcal{L}_r(w_i)$ and $\delta\mathcal{L}_f(w_i)$ can be upper bounded by $L\|\delta w\| \to 0$, for small perturbations. So with the small upper bound we can approximately say that both the quantities are very close to each other.

So we define an approximate version of f_i as follows: $\tilde{\mathrm{f}}_i(\mathrm{H}_r) = \frac{1}{2}(\delta w)_i^\top \mathrm{H}_r(w^\star)(\delta w)_i - \nabla_f(w^\star)^\top(\delta w)_i - \delta\mathcal{L}_f(w_i)$. Our final objective becomes:

$$\tilde{\Psi}(\mathrm{H}_r) = \frac{1}{m}\sum_{i=1}^{m}\left(\tilde{\mathrm{f}}_i(\mathrm{H}_r)\right)^2$$

Clearly the H_r is positive semi definite (PSD) for any convex loss functions. Based on this observation, we formulate the following optimization as a Semi Definite Program (SDP) as follows:

$$\begin{aligned}&\text{minimize} && \tilde{\Psi}(X)\\ &\text{subject to} && \mathrm{X} \succeq 0\end{aligned} \tag{3}$$

Since we are approximating the value of $\delta\mathcal{L}_r(w_i)$ instead of using the actual ground truth value, we anticipate that the solution to optimization problem Eq. (3) will be approximately close to the true retained Hessian H_r. In fact, we can bound the error between the true and estimated Hessian using the following lemma.

Lemma 1. *Consider choosing $\delta w \in \mathbb{R}^d$ where each element $\delta w(j)$ of is sampled from $\mathcal{N}(0,1)$. Assuming that the solution of the optimization Eq. (3) converges to $\hat{H}_r$, then the frobenius norm of the difference between the Hessian H_r (the actual ground truth Hessian with respect to $\mathcal{D}_r$) and $\hat{H}_r$ can be upper bounded as:*

$$\|\Delta H_r\|_F \leq \frac{2\epsilon\sqrt{d}}{(2+d)}$$

where ϵ is the upper bound on the approximation error of $\delta\mathcal{L}_r(w_i)$. Mathematically: $|\delta\mathcal{L}_r(w_i) - \delta\mathcal{L}_f(w_i)| \leq \epsilon\ \forall i$

Proof. From the definition of $\Psi(\mathrm{H})$:

$$\Psi(\mathrm{X}) = \mathop{\mathbb{E}}_{\delta w \sim \mathcal{N}(\mathbf{0},\mathbf{I})}\left[(\frac{1}{2}\delta w^\top \mathrm{X}\delta w + \nabla_r^\top \delta w - \delta\mathcal{L}_r)^2\right] \tag{4}$$

By neglecting higher order terms in the taylor approximation we can say, $\delta\mathcal{L}_r \approx \frac{1}{2}\delta w^\top H_r \delta w + \nabla_r^\top \delta w$. Substituting

$\delta\mathcal{L}_r$ from Equation 4:

$$\begin{aligned}\Psi(\mathrm{X}) &= \mathbb{E}_{\delta w\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[(\frac{1}{2}\delta w^\top \mathrm{X}\delta w - \frac{1}{2}\delta w^\top \mathrm{H}_r\delta w)^2\right]\\ &= \mathbb{E}_{\delta w\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[(\frac{1}{2}\delta w^\top (\mathrm{X}-\mathrm{H}_r)\delta w)^2\right]\\ &= \frac{1}{2}\mathrm{trace}(\mathrm{M}^2) + \frac{1}{4}\mathrm{trace}(\mathrm{M})^2\end{aligned}$$

where, we define $\mathrm{M} = (\mathrm{X} - \mathrm{H}_r)$. So, clearly the minimizer of $\Psi(\mathrm{X})$ is at $\mathrm{M} = 0$ or $\mathrm{X} = \mathrm{H}_r$. However it is the ideal case, where we do not approximate $\delta\mathcal{L}_r$. In our algorithm, we are minimizing and approximate objective $\tilde{\Psi}(\mathrm{X})$. Also from the definition we can say $\tilde{\mathrm{f}}_i(X) = \mathrm{f}_i(\mathrm{X}) + (\delta\mathcal{L}_r(w_i) - \delta\mathcal{L}_f(w_i))$. Since we assume that $|\delta\mathcal{L}_r(w_i) - \delta\mathcal{L}_f(w_i)| \leq \epsilon\ \forall i$, we can derive the following inequality:

$$\begin{aligned}&(\mathrm{f}_i(\mathrm{X}) - \epsilon) \leq \tilde{\mathrm{f}}_i(X) \leq (\mathrm{f}_i(\mathrm{X}) + \epsilon)\\ \implies &\frac{1}{m}\sum_{i=1}^{m}(\mathrm{f}_i(\mathrm{X}-\epsilon))^2 \leq \tilde{\Psi}(\mathrm{X}) \leq \frac{1}{m}\sum_{i=1}^{m}(\mathrm{f}_i(\mathrm{X}+\epsilon))^2\end{aligned}$$

Using this definition and using the derived term for $\Psi(\mathrm{X})$, we can derive the following:

$$\begin{aligned}&\tilde{\Psi}(\mathrm{X}) \leq \frac{1}{2}\mathrm{trace}(\mathrm{M}^2 + 2\epsilon\mathrm{M}) + \frac{1}{4}\mathrm{trace}(\mathrm{M})^2\\ &\frac{1}{2}\mathrm{trace}(\mathrm{M}^2 - 2\epsilon\mathrm{M}) + \frac{1}{4}\mathrm{trace}(\mathrm{M})^2 \leq \tilde{\Psi}(\mathrm{X})\end{aligned}$$

By seperately taking derivatives of the upper and lower bounds above, if we set it to 0, we get the following bound on the minimizer M (**details in the supplementary**).

$$-\frac{2\epsilon}{(2+d)}\mathrm{I}_d \leq \mathrm{M} \leq \frac{2\epsilon}{(2+d)}\mathrm{I}_d$$

where, $\mathrm{I}_d \in \mathbb{R}^{d\times d}$ is the identity matrix. This inequality implies that the if the solution of optimization 3 is $\hat{\mathrm{H}}_r$, then

$$\mathrm{H}_r - \frac{2\epsilon}{(2+d)}\mathrm{I}_d \leq \hat{\mathrm{H}}_r \leq \mathrm{H}_r + \frac{2\epsilon}{(2+d)}\mathrm{I}_d$$

As a result we can conclude:

$$\|\hat{\mathrm{H}}_r - \mathrm{H}_r\| = \|\Delta\mathrm{H}_r\|_F \leq \frac{2\epsilon\|\mathrm{I}_d\|_F}{(2+d)}$$

Since $\|\mathrm{I}_d\|_F = \sqrt{d}$, we conclude the proof. □

Implications of Lemma 1: The lemma provides an upper bound on the Frobenius norm between the true and estimated Hessians, characterized by two parameters: ϵ and d. When ϵ is smaller, the upper bound decreases, leading to a better approximation of the retain Hessian. This is intuitive, as ϵ represents the approximation error of the loss difference. Additionally, increasing the feature dimension d causes the upper bound to approach zero. Specifically, the norm decreases inversely with $\sqrt{d}$. In other words, as the matrix size grows, the upper bound on the difference becomes smaller. Importantly, we do not make any assumptions about the linearity of the model in proving this lemma. The bound suggests that for high-dimensional scenarios (such as in deep models), estimating the Hessian accurately may be feasible, making this method worth exploring. However, storing and inverting a large Hessian is computationally impractical. Fortunately, various methods exist for Hessian approximation, such as diagonalization [32] or linearizing deep models using Hessian-vector products [13]. Integrating these approximation techniques with our approach could open up promising avenues for future research.

Theorem 1. *Suppose that $\forall(x_i, y_i) \in \mathcal{D}$, $w \in \mathbb{R}^d$: $\|\nabla\ell(w^\top x_i, y_i)\| \leq C$. Suppose that the second derivative of ℓ is γ-lipschitz and $\|x_i\|_2 \leq 1$ for all $(x_i, y_i) \in \mathcal{D}$, and the result of optimization Eq. (3) is $\hat{H}$. Then:*

$$\begin{aligned}\|\nabla\mathcal{L}(w_{uf}, \mathcal{D}_r)\|_2 &\leq \gamma(n-n_f)\|\hat{H}_r^{-1}\nabla_f\|_2^2\\ &\leq \frac{4\gamma C^2 n_f^2(n-n_f)}{\left[\lambda(n-n_f) - \frac{2\epsilon}{(2+d)}\right]^2}\end{aligned}$$

where n and n_f denote the number of total training samples and the number of samples to be forgotten respectively.

Proof. This proof is inspired by [15], and based on **Theorem 4** of the paper. This bound says that upon forgetting n_f samples from the dataset, if the resulting model become w_{uf} then the norm of the gradient with respect to this model on the remaining dataset can be upper bounded as follows:

$$\|\nabla\mathcal{L}(w_{uf}, \mathcal{D}_r)\|_2 \leq \gamma(n-n_f)\|\mathrm{H}_r^{-1}\nabla_f\|_2^2$$

Now this H is the actual retain hessian while our estimate is $\hat{\mathrm{H}}_r = \mathrm{H}_r + \Delta\mathrm{H}_r$. From the definition of loss $\mathcal{L}$, we know that after removing n_f samples the loss becomes $\lambda(n-n_f)$-strongly convex. As a result we get $\|\mathrm{H}_r\|_2 \geq \lambda(n-n_f)$. Now we can apply triangle inequality and the upper bound from Lemma 1 as follows:

$$\begin{aligned}\lambda(n-n_f) \leq \|\mathrm{H}_r\|_2 &\leq \|\hat{\mathrm{H}}_r\|_2 + \|\Delta\mathrm{H}_r\|_2\\ &\leq \|\hat{\mathrm{H}}_r\|_2 + \left\|\frac{2\epsilon}{(2+d)}\mathrm{I}_d\right\|_2\\ &= \|\hat{\mathrm{H}}_r\|_2 + \frac{2\epsilon}{(2+d)}\end{aligned}$$

which implies,

$$\begin{aligned}&\|\hat{\mathrm{H}}_r\|_2 \geq \lambda(n-n_f) - \frac{2\epsilon}{(2+d)}\\ \implies &\|\hat{\mathrm{H}}_r\|_2^{-1} \leq \frac{1}{\lambda(n-n_f) - \frac{2\epsilon}{(2+d)}}\end{aligned}$$

Also, from **Theorem 4** of [15], we know $\|\nabla_f\| \leq 2Cn_f$. So,

$$\|\hat{\mathrm{H}}_r^{-1}\nabla_f\|_2^2 \leq \|\hat{\mathrm{H}}_r^{-1}\|_2^2\|\nabla_f\|_2^2 \leq \frac{4C^2n_f^2}{(\lambda(n-n_f) - \frac{2\epsilon}{(2+d)})^2}$$

$$\implies \|\nabla\mathcal{L}(w_{uf}, \mathcal{D}_r)\|_2 \leq \frac{4\gamma C^2n_f^2(n-n_f)}{(\lambda(n-n_f) - \frac{2\epsilon}{(2+d)})^2}$$

Hence we conclude the proof of Theorem 1. □

Implications of Theorem 1: The leftmost term in the theorem's inequality essentially represents the norm of the gradient with respect to the unlearned model on the remaining data. Successful unlearning, as suggested by the parameter indistinguishability (see Sec. 3.2), should lead this norm to approach 0. Examining the upper bound, we observe that for a fixed size d of the Hessian, it becomes tighter with an increase in the number of remaining data, as the term is inversely proportional to the remaining data size. We validate this phenomenon through experimentation, where we observe a decline in unlearning performance as the number of samples to be forgotten increases. Additionally, the upper bound decreases as we significantly increase the Hessian dimension d. This finding aligns with Lemma 1, which asserts that for large d, the estimated and true Hessians closely resemble each other, indicating effective unlearning.

5. Experiments

Datasets. To demonstrate the efficacy of our proposed algorithms for the source-free unlearning scenario, we use four standard benchmark classification datasets: CIFAR-10 [19], CIFAR-100 [19], Stanford Dogs [18], and CalTech-256 [14]. CIFAR-10 is a dataset consisting of 60,000 RGB images in 10 different classes, with 6,000 images per class. CIFAR-100 is similar to CIFAR-10, but with 100 classes containing 600 images each, providing a more granular classification challenge. Stanford Dogs contains 20,580 images of 120 different breeds of dogs, making it ideal for fine-grained classification tasks. CalTech-256 comprises 30,607 images across 256 object categories, offering a diverse set of images for comprehensive object recognition research.

Implementation details. Since we perform zero-shot unlearning for linear classifiers, we use a ResNet-18 [16] architecture pre-trained on ImageNet [6] as our feature extractor. Using these features, we train a linear classifier and then discard the data. During unlearning, we only use the trained linear model and the data to be forgotten. For the experiments with neural networks we used Mixed-Linear neural network [13] and we linearized the last few layers. We used ResNet-18 [16] and the datasets given above for these experiments. We randomly sample up to 10% of the training data as the forget data. All experiments were performed on a single NVIDIA-RTX 3090 GPU.

Baseline metrics. The main baseline for any Machine Unlearning (MU) methods is the parameter indistinguishability between the retrained and unlearned models. A successful unlearning algorithm should emulate the performance of a model that was never exposed to the forget data, having been trained solely on the remaining data. In this context, we evaluate the classification accuracy of the model on the following datasets: (i) test data, (ii) remaining training data, and (iii) forget data. Additionally, we assess the Membership Inference Attack (MIA) score of the models, as proposed in the prior work [20]. This score indicates whether a sample was originally part of the training set. After forgetting certain samples, we check their MIA scores using the unlearned model. An MIA score close to 50% signifies successful unlearning, as it indicates that the unlearned model cannot distinguish whether the forget data came from the training distribution or the test distribution.

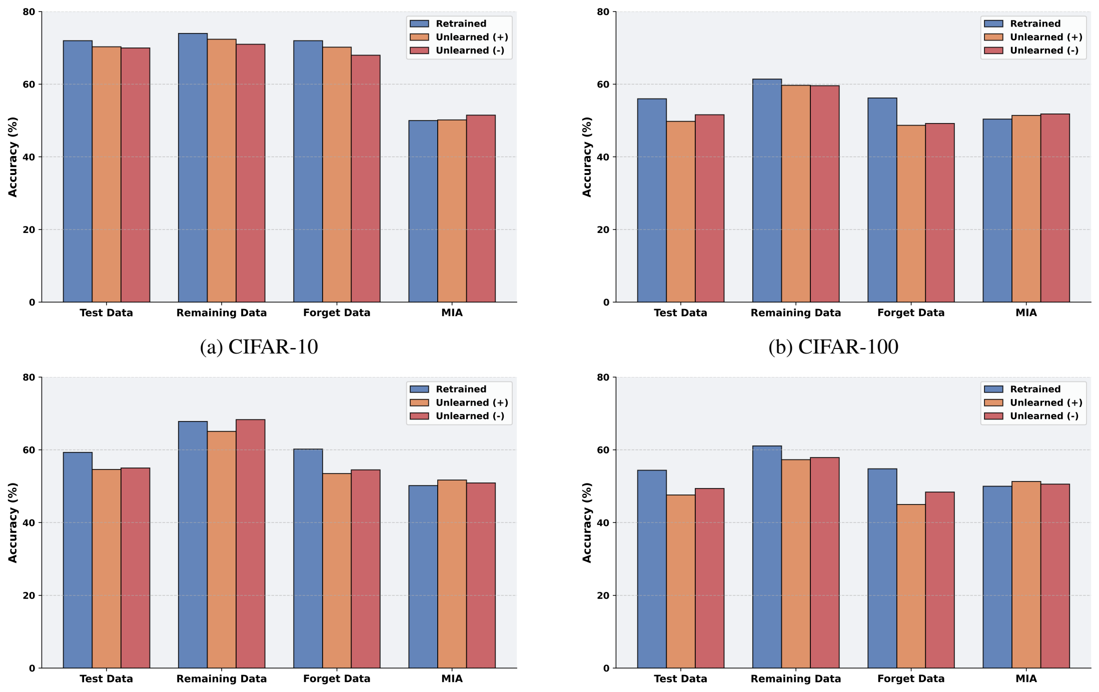
Baseline models. Unlearned models using our proposed algorithm are compared with two types of models: (a) A model retrained from scratch using the remaining training data (Retrained) and (b) An unlearned model that has been unlearned using the exact Hessian computed from the remaining data (Unlearned(+)). Since we estimate the retain Hessian without accessing the remaining data, our primary objective is to closely mimic the performance of the model described in (a) (marked with cyan color for all the tables) using the baseline metrics. Since we do not need the training data during unlearning we refer the unlearned model using *our proposed algorithm as **(Unlearned(-))***. We explore these model's performances using both linear and mixed-linear classifiers for all the datasets.

5.1. Comparison of baseline metrics on different datasets using linear classifiers

We compare the performance of the Retrained, Unlearned(+), and Unlearned(-) models on all four datasets (Fig. 1) by selecting 10% of the training samples as forget data. We use linear classifier and quadratic loss as the convex loss function for all cases. The results are presented as bar plots for all these scenarios. As theoretically expected, the performance of Unlearned(+) closely mimics that of the Retrained model. As per the main results, in all cases, the performance of Unlearned(-) closely matches that of the Unlearned(+) model, which aligns with our theoretical bounds.

5.2. Effects of percentage of the forget data size

To investigate the influence of forget data size, we vary the proportion of randomly selected data for forgetting within the training set while maintaining consistency across all other factors. According to Theorem 1, it becomes apparent that the quantity of forgotten samples significantly influences the optimization process. As we can see in Tab. 1,



(a) CIFAR-10 (b) CIFAR-100

(c) StanfordDogs (d) Caltech256

Figure 1. Performance comparison of the proposed methods across different datasets: CIFAR-10, CIFAR-100, StanfordDogs, and Caltech256. We randomly select 10% of the entire training data as forget samples. Each figure illustrates the effectiveness of the optimization strategies in handling the forgetting of samples, as evidenced by the close performance of models *Unlearned(+)* and *Unlearned(-)*. As expected, our method also yields results comparable to retraining from scratch (*Retrained*), suggesting successful unlearning.

increasing the number of forget samples negatively impacts performance. Specifically, when 5% of the training data is chosen randomly for forgetting, the disparity between the retrained model with the remaining data and the model updated using our approach becomes negligible. However, with an increase in the percentage of forget data, the gap between these two models widens considerably. This result perfectly matches the bound we provide in Theorem 1.

Method		Test	Remaining	Forget	MIA
Retrained		73%	75%	73%	50%
Unlearned (-)	15%	59%	60%	59%	55.8%
Performance Gap		**14%**	**15%**	**14%**	**5.8%**
Retrained		72%	74%	72%	50%
Unlearned (-)	10%	70%	71%	68%	51.4%
Performance Gap		**2%**	**3%**	**4%**	**1.4%**
Retrained		73%	74%	73%	49.4%
Unlearned (-)	5%	73%	74%	73%	49.4%
Performance Gap		**0%**	**0%**	**0%**	**0%**

Table 1. The effect of the proportion of randomly selected data from the CIFAR-10 training dataset for forgetting. As the number of forget data samples increases, the difference in performance between the Retrained and Unlearned(-) models also increases. The second column indicates the percentage of the selected forgetting data.

5.3. Effects of the number of perturbations

Method		Test	Remaining	Forget	MIA
Retrained		72%	74%	72%	50%
Unlearned (-)	250	57%	58%	57%	56.2%
Performance Gap		**15%**	**16%**	**15%**	**6.2%**
Unlearned (-)	500	70%	71%	68%	51.4%
Performance Gap		**2%**	**3%**	**4%**	**1.4%**
Unlearned (-)	1000	72%	74%	71%	49%
Performance Gap		**0%**	**0%**	**1%**	**1%**

Table 2. The effect of the number of perturbations randomly selected from a Gaussian distribution for the CIFAR-10 dataset. The second column indicates the number of perturbations used to approximate the Hessian using our method. Increasing the number of perturbations positively influences unlearning performance.

For this experiment, we perform unlearning with linear classifiers by varying the number of perturbations. As demonstrated in the proof of Lemma 1, the convergence of our optimization toward the true retain Hessian is strongly influenced by the objective function, which is formulated as the expectation of a perturbed random variable. Consequently, to better capture the expected value, a larger number of perturbations is required, which, in turn, positively impacts unlearning performance. This effect is clearly illustrated in Table 2, where we observe that a higher number of perturbations consistently leads to improved outcomes.

5.4. Effects of the L2 regularization

Method		Test	Remaining	Forget	MIA
Retrained		72%	74%	72%	50%
Unlearned (-)	0	70%	71%	68%	51.4%
Performance Gap		**2%**	**3%**	**4%**	**1.4%**
Unlearned (-)	0.0005	71%	72%	71%	49.8%
Performance Gap		**1%**	**2%**	**1%**	**0.2%**
Unlearned (-)	0.001	72%	73%	72%	50.9%
Performance Gap		**0%**	**1%**	**0%**	**0.9%**

Table 3. The impact of the regularization parameter λ on the unlearning algorithm. Increasing λ leads to improved unlearning performance, consistent with our claim in Theorem 1.

The theoretical upper bound on the norm in Theorem 1 is clearly proportional to $\frac{1}{\lambda^2}$, with λ representing the regularization parameter. Consequently, as demonstrated in Tab. 3, increasing the value of λ leads to a reduction in the performance gap between our unlearned model and the retrained model.

5.5. Experiments on mixed linear networks

To show the applicability of our approach to the neural networks, we consider using mixed linear networks for unlearning [13]. We select the last few layers of ResNet-18 to linearize and train it with CIFAR-100. We randomly select 10%, 15%, and 20% of data to be forgotten from the trained model and apply our optimization function to approximate the remaining data Hessian at the trained model. The results are listed in Tab. 4. As can be seen, our method can approximate the exact remaining hessian and can perform significantly well even without having the remaining data. We also conduct experiments on other datasets mentioned in Sec. 5 [Datasets](See supplementary).

Method		Test	Remaining	Forget	MIA
Retrained		62.2%	65.7%	62.6%	50%
Unlearned (+)	20%	60.1%	63.4%	59.3%	50.4%
Unlearned (-)		60%	70%	63.2%	50.8%
Performance Gap		**0.1%**	**6.6%**	**3.9%**	**0.4%**
Retrained		63.1%	68.9%	63.3%	50.2%
Unlearned (+)	15%	61.9%	67.9%	62.2%	50.7%
Unlearned (-)		61.4%	70.1%	62.2%	51.7%
Performance Gap		**0.5%**	**2.2%**	**0.0%**	**1.0%**
Retrained		63.7%	72%	63.9%	50%
Unlearned (+)	10%	63.3%	72.5%	64.9%	51%
Unlearned (-)		62.8%	70.1%	61.2%	52.3%
Performance Gap		**0.5%**	**2.4%**	**3.7%**	**1.3%**

Table 4. We demonstrate the applicability of our method to neural networks using mixed linear networks. The method performs considerably well even without access to remaining data. Experiments were conducted on the CIFAR-100 dataset with a ResNet-18 model, linearizing the last few layers.

5.6. Comparison with other source-free unlearning approaches

We compared our method with other unlearning approaches closely related to our work. For these experiments, we utilized the last layer of the ResNet-18 architecture as a linear classifier on the CIFAR-10 dataset, randomly selecting 10% of the data to be forgotten across all tests. As shown in Tab. 5, our approach outperforms existing source-free methods significantly in terms of performance on remaining, forget, and test data. Specifically, **NegGrad** [12] fine-tunes using only the forget data by applying gradient descent to increase the loss for forget data. The **Random Labels** [12] method randomly reassigns labels (excluding the actual class) to forget samples and fine-tunes the model with this modified data. **JiT** [9] employs Lipschitz regularization to maintain stable model outputs across perturbed data samples for unlearning. The **Adversarial** [3] approach combines adversarial examples with weight importance metrics to preserve performance on remaining data while applying gradient ascent on the forget data.

Method	Test Data	Remaining Data	Forget Data	MIA
Retrained	72%	74%	72%	50%
NegGrad	51.9%	53.2%	51.2%	48%
Random Labels	20.6%	21.6%	21.4%	47%
JiT	52.1%	53.1%	51.1%	49.1%
Adversarial	51.5%	52.7%	51.0%	50.0%
Unlearned (-)	**70%**	**71%**	**68%**	**51.4%**

Table 5. Comparison of existing source-free unlearning methods with our proposed method (Unlearned (-)). Our method significantly outperforms others. Experiments are conducted on the CIFAR-10 dataset using a linear classifier.

6. Conclusion

In this paper, we introduce and evaluate a novel unlearning algorithm tailored for linear and mixed linear classifiers, specifically targeting scenarios where the original training data is unavailable during the unlearning process. Our algorithm is a general-purpose method adaptable to a wide range of convex loss functions, enabling its application across diverse contexts where different convex loss functions are employed. This flexibility makes it a versatile tool for unlearning in various machine learning applications. We establish robust theoretical bounds for our algorithm, providing assurances of its reliability and effectiveness in unlearning tasks. These theoretical guarantees offer a solid basis for understanding the algorithm's behavior and performance. Additionally, we examine the implications of these bounds and validate the practical efficacy of our algorithm through extensive experimental evaluations. The results demonstrate that the proposed algorithm performs exceptionally well, confirming its theoretical advantages and highlighting its potential for real-world applications.

7. Acknowledgment

This work was supported in part by the NSF CAREER Award CCF-2144927, NSF Award CCF-2008020, DURIP N000141812252, the UCR OASIS Fellowship, and the Amazon Research Award.

References

[1] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 141–159. IEEE, 2021. 2

[2] Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *2015 IEEE symposium on security and privacy*, pages 463–480. IEEE, 2015. 2

[3] Sungmin Cha, Sungjun Cho, Dasol Hwang, Honglak Lee, Taesup Moon, and Moontae Lee. Learning to unlearn: Instance-wise unlearning for pre-trained classifiers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 11186–11194, 2024. 2, 3, 8

[4] Ruizhe Chen, Jianfei Yang, Huimin Xiong, Jianhong Bai, Tianxiang Hu, Jin Hao, Yang Feng, Joey Tianyi Zhou, Jian Wu, and Zuozhu Liu. Fast model debias with machine unlearning. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[5] Vikram S Chundawat, Ayush K Tarun, Murari Mandal, and Mohan Kankanhalli. Zero-shot machine unlearning. *IEEE Transactions on Information Forensics and Security*, 2023. 1, 2

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 6

[7] Yonatan Dukler, Benjamin Bowman, Alessandro Achille, Aditya Golatkar, Ashwin Swaminathan, and Stefano Soatto. Safe: Machine unlearning with shard graphs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17108–17118, 2023. 2

[8] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014. 2

[9] Jack Foster, Kyle Fogarty, Stefan Schoepf, Cengiz Öztireli, and Alexandra Brintrup. Zero-shot machine unlearning at scale via lipschitz regularization. *arXiv preprint arXiv:2402.01401*, 2024. 3, 8

[10] Rohit Gandikota, Joanna Materzynska, Jaden Fiotto-Kaufman, and David Bau. Erasing concepts from diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2426–2436, 2023. 2

[11] General Data Protection Regulation GDPR. General data protection regulation. *URL: https://gdpr-info. eu/[accessed 2020-11-21]*, 2018. 2

[12] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9304–9312, 2020. 1, 2, 8

[13] Aditya Golatkar, Alessandro Achille, Avinash Ravichandran, Marzia Polito, and Stefano Soatto. Mixed-privacy forgetting in deep networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 792–801, 2021. 2, 3, 5, 6, 8

[14] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. Technical report, California Institute of Technology, 2007. 6

[15] Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified data removal from machine learning models. In *Proceedings of the 37th International Conference on Machine Learning*. JMLR.org, 2020. 1, 2, 3, 5, 6

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6

[17] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018. 3

[18] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, 2011. 6

[19] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, University of Toronto, Toronto, ON, Canada, 2009. 6

[20] Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. Towards unbounded machine unlearning. *Advances in Neural Information Processing Systems*, 36, 2024. 1, 6

[21] Guihong Li, Hsiang Hsu, Radu Marculescu, et al. Machine unlearning for image-to-image generative models. *arXiv preprint arXiv:2402.00351*, 2024. 2

[22] Zhiyuan Li, Ruosong Wang, Dingli Yu, Simon S Du, Wei Hu, Ruslan Salakhutdinov, and Sanjeev Arora. Enhanced convolutional neural tangent kernels. *arXiv preprint arXiv:1911.00809*, 2019. 3

[23] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International conference on machine learning*, pages 6028–6039. PMLR, 2020. 2

[24] Alessandro Mantelero. The eu proposal for a general data protection regulation and the roots of the 'right to be forgotten'. *Computer Law & Security Review*, 29(3):229–235, 2013. 1

[25] Seth Neel, Aaron Roth, and Saeed Sharifi-Malvajerdi. Descent-to-delete: Gradient-based methods for machine unlearning. In *Algorithmic Learning Theory*, pages 931–962. PMLR, 2021. 2

[26] Ayush K Tarun, Vikram S Chundawat, Murari Mandal, and Mohan Kankanhalli. Fast yet effective machine unlearning. *IEEE Transactions on Neural Networks and Learning Systems*, 2023. 1

[27] Paul Voigt and Axel Von dem Bussche. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 10(3152676):10–5555, 2017. 1

[28] Alexander Warnecke, Lukas Pirch, Christian Wressnegger, and Konrad Rieck. Machine unlearning of features and labels. *arXiv preprint arXiv:2108.11577*, 2021. 3

[29] Ga Wu, Masoud Hashemi, and Christopher Srinivasa. Puma: Performance unchanged model augmentation for training data removal. In *Proceedings of the AAAI conference on artificial intelligence*, pages 8675–8682, 2022. 2

[30] Yinjun Wu, Edgar Dobriban, and Susan Davidson. Deltagrad: Rapid retraining of machine learning models. In *International Conference on Machine Learning*, pages 10355–10366. PMLR, 2020. 2

[31] Yuanshun Yao, Xiaojun Xu, and Yang Liu. Large language model unlearning. *arXiv preprint arXiv:2310.10683*, 2023. 2

[32] Zhewei Yao, Amir Gholami, Sheng Shen, Mustafa Mustafa, Kurt Keutzer, and Michael Mahoney. Adahessian: An adaptive second order optimizer for machine learning. In *proceedings of the AAAI conference on artificial intelligence*, pages 10665–10673, 2021. 5

[33] Jingwen Ye, Yifang Fu, Jie Song, Xingyi Yang, Songhua Liu, Xin Jin, Mingli Song, and Xinchao Wang. Learning with recoverable forgetting. In *European Conference on Computer Vision*, pages 87–103. Springer, 2022. 1