CAFFEINE: MACHINE UNLEARNING VIA CURVATURE-AWARE GRADIENT ASCENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Deleting data from a trained model without retraining requires an update that cancels the deleted data's contribution. Influence functions specify this update via the inverse Hessian, but evaluating it requires iterative passes over the training data, which is often no longer available when a deletion request arrives. Existing approximations read the retained data, restrict the change to the final layer, or drop curvature rescaling altogether. We present Caffeine, a source-free unlearning algorithm that operates as a curvature-aware gradient ascent. Since the update moves along the gradient, it needs the curvature strictly along that one direction—a single scalar that two forget-set gradients evaluated a small step apart determine without any exact second-order computation or retained data. Caffeine measures this scalar at each saved training checkpoint and aggregates the resulting curvature-scaled steps into a single update to every layer of the deployed model. Across class and random-subset deletion on five image and text datasets, Caffeine is the only approximate method that both forgets and preserves accuracy without retained data, running up to $25.3\times$ faster than retraining on image tasks.

1 INTRODUCTION

Deletion requests are now a routine operation on trained models: a user withdraws consent under the *Right to be Forgotten* [2, 30, 32], a data source turns out to be mislicensed, or a regulator orders a category of records removed. The exact response—retraining on the retained data—costs a full training run per request, and requests keep coming. Approximate unlearning avoids retraining: it updates the trained parameters to approximate what retraining would have produced. Influence functions [20] tell us what that update should be: the gradient of the loss on the deleted data, rescaled by the inverse Hessian of the training loss. The gradient costs one backward pass. The inverse Hessian cannot be formed for a network with N parameters, and the iterative estimators that avoid forming it [13, 20] run over the training data, which in practice is often *no longer available* when a deletion request arrives. Thus, unlearning systems should operate under the source-free setting [2]: the update is computed from the forget set, the deployed model, and its training checkpoints, without reading the retained data.

Existing approximations meet only part of this requirement. Kronecker-factored and low-rank approximations make the inverse Hessian tractable but compute it from the retained data [13, 24]. Gradient-ascent and distillation methods discard the rescaling altogether and use retained data to judge how far to move; without a calibrated step they either erase too little or damage retained accuracy [22, 38]. Trajectory methods precompute and store one model-sized vector for every training sample, so their memory is proportional to the training set rather than to the deletion, and on deep networks it runs out before the first deletion is served [30]. Source-free unlearning avoids retained data but touches only the final layer [2]. It changes how the model labels the deleted data but not what it learned from it: every earlier layer still encodes its features, and the model retains what it was asked to forget.

In this paper, we present Caffeine that closes this gap. Caffeine's design rests on three key observations. *First*, the update needs far less than the full Hessian. The gradient says which way to move; the inverse Hessian says how far: a short step where the loss curves sharply, a long one where it is flat. It records this for every direction at once, which is why it is $N \times N$. But the update moves along one direction only, the gradient, so it needs the curvature along that direction alone—a single number

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

K—and the right step is the gradient divided by K. Thus Caffeine's approximation is the same curvature in every direction. *Second*, K can be measured with two gradients. Curvature is the rate at which the gradient changes, so Caffeine computes the forget-set gradient, takes a small step, and computes it again. Projecting the change between the two onto the step direction, per unit of step, gives K: no Hessian, and no retained data. *Third*, one step is not enough. The deleted data shaped the parameters at every stage of training, under a different K each time. Caffeine therefore repeats the measurement at each saved checkpoint and combines the resulting steps into a single update.

The result is a curvature-aware gradient ascent derived from the training trajectory. It reads no retained data, updates every parameter, sizes every step by the curvature at its own checkpoint, and needs memory for one model and two gradients at a time. None of the methods above offers all four; Caffeine offers all four. The hardest deletion is an *entire class*: its features are encoded in every layer, so a method that touches part of the network leaves them in place, and a method that erases them without calibration takes the neighboring classes with them.

We evaluate Caffeine on class deletion (one to twenty classes) and random-subset deletion (10% of the training set) across CIFAR-10, TinyImageNet, a face-identity dataset, and two text sentiment datasets, against retraining and four approximate methods (SFMU, HFOL, GradTao, SCRUB). Every baseline either fails to forget or forgets at the cost of 14.35 to 50.98 points of test accuracy. Caffeine is the only approximate method that does both on every dataset: forget accuracy within 1.37 points of retraining's and test accuracy within 4.35 points on images and 5.33 on text, while running 12.9–$25.3\times$ faster than retraining on images and over $2\times$ faster on text, with up to $1.65\times$ less peak memory. On random-subset deletion, a membership-inference attack separates the forgotten data from unseen data less than it does for any approximate baseline.

In summary, our contributions are:

- Caffeine, a source-free unlearning algorithm that reaches every layer and calibrates its step size, with no retained data and no Hessian.
- A Hessian-free curvature estimate: the change in the forget-set gradient over a small probe step, obtained from two gradient evaluations and requiring no second-order computation.
- A trajectory aggregation scheme that evaluates the curvature-scaled ascent step at each saved checkpoint and combines these into a single update, with cost independent of the retain set.

2 Problem Formulation

Setup. Let $D = \{z_1, \ldots, z_n\}$ be a training dataset, where each sample z_i consists of an input (with features) and a label. A model is a vector of weights $\theta \in \mathbb{R}^N$, and the loss $L(z_i, \theta)$ measures the model's error on z_i. Training minimizes the average loss over D to obtain the parameters,

$$\theta^* \;=\; \arg\min_{\theta} \; \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta). \tag{1}$$

Unlearning. A deletion request specifies a forget set $D_f \subseteq D$ such that the remainder is known as the retain set, $D_{ret} = D \setminus D_f$. We consider two types of deletion request: *class deletion*, where D_f consists of all samples that have the same label, and *random-subset deletion*, where D_f consists of randomly selected samples. We do not evaluate single-record deletion separately. The brute force method to perform a deletion is calculating θ^*_{ret} (equation 1) on the retain set, D_{ret}. However, the computation cost scales with the size of the retain set, $|D_{ret}|$. A more efficient alternative is to update θ^* using the gradient $\nabla_\theta L(z, \theta) \in \mathbb{R}^N$ of each sample $z \in D_f$, and the Hessian $H \in \mathbb{R}^{N \times N}$ of the training loss.

Traditional influence functions (Section 3.1) estimate the unlearned parameters by moving along the inverse Hessian-vector product: $H^{-1} \nabla_\theta L(z, \theta)$. Furthermore, because the curvature of the loss landscape differs across data subsets ($H \neq H_{ret} \neq H_f$), substituting the inaccessible global Hessian with the forget-set Hessian does not mathematically yield an exact geometric inverse of the training influence. Rather than estimating H^{-1}, Caffeine keeps the form of the influence update—a forget-set gradient rescaled by inverse curvature—and formulates unlearning as *curvature-aware gradient ascent*. The gradient requires one backpropagation pass over D_f and gives the ascent direction that increases the forget-set loss. To calibrate the step size along that direction, Caffeine approximates

the curvature as isotropic ($H \approx K\mathbf{I}$ for a scalar K) and measures K on D_f alone, so the step is set without D_{ret}.

Problem. *Given θ^* trained on D and a forget set $D_f \subseteq D$, compute θ_u such that: (i) θ_u approximates θ^*_{ret}, i.e. its accuracy and membership-inference AUC on D_f are close to those of θ^*_{ret} (Caffeine is approximate and does not certify (ϵ, δ)-indistinguishability [15]); (ii) Accuracy on D_{ret} is preserved; and (iii) The computation uses only θ^*, gradients on D_f, and the C checkpoints and learning rates saved during training, but not the retain set, D_{ret}.*

3 Methodology: Caffeine

We now present Caffeine, an approximate unlearning algorithm designed to avoid the limitations of its predecessors: (1) It does not retrain on the retain set, D_{ret}. (2) It does not form or invert the Hessian, avoiding its $O(N^2)$ memory and $O(|D| \cdot N^2 + N^3)$ time. (3) Instead of a validation set drawn from D_{ret}, it uses only the deployed model, the saved checkpoints, and the forget set (storage cost discussed in Section 2). (4) It aims to erase D_f while preserving model quality.

At each training checkpoint, Caffeine estimates how far and in which direction the model's parameters must shift to remove D_f's effect on the loss. These per-checkpoint shifts are combined into a single vector (Section 3.2.3), which is added to the deployed model to produce the unlearned model.

In the rest of this section, we first define the shift via the Hessian (Section 3.1), then show how Caffeine estimates it without ever computing or inverting that Hessian (Sections 3.2.1 to 3.2.3).

3.1 Measuring Influence via Hessian

An *influence function* estimates the effect a sample has on the trained model's parameters [20]. For influence estimation, we first calculate the Hessian of the model's loss, averaged over the training set:

$$H_{\theta^*} \;=\; \frac{1}{n}\sum_{i=1}^{n} \nabla^2_\theta L(z_i, \theta^*), \tag{2}$$

where $\nabla_\theta L(z_i, \theta^*)$ is the first-order derivative (*Gradient*) of the loss w.r.t. parameters (weights) θ^*, and $\nabla^2_\theta L(z_i, \theta^*)$ is the second-order derivative (*Hessian*). Intuitively, the Hessian is a map of how sharply the loss landscape bends around θ^* in every direction at once. High curvature means θ^* sits in a sharp minimum, where even a small step causes a large increase in loss; low curvature means a flat region, where the parameters can move far while the loss changes little. The influence of a forgotten sample $z_f \in D_f$ on the parameters θ^* is calculated as,

$$\mathcal{I}(z_f) \;=\; -\,H^{-1}_{\theta^*}\,\nabla_\theta L(z_f, \theta^*), \tag{3}$$

where $H^{-1}_{\theta^*}$ is the inverse of the Hessian. The inverse Hessian rescales the gradient of the forgotten sample differently in each direction, so it sets both the *direction* in which the model must move to forget z_f and the *step size* in that direction. To first order, the influences of the samples in a forget set add, and the deletion is applied in a single step:

$$\theta_u \;=\; \theta^* - \frac{1}{n}\sum_{z_f \in D_f} \mathcal{I}(z_f). \tag{4}$$

Equation (3) is a first-order approximation that assumes a strictly convex loss at θ^*, and inverting the Hessian has a time complexity of $O(N^3)$ for a model with N weights. Caffeine avoids both the Hessian and its inversion: it first computes the forget set's gradient and then rescales it by a step size determined by the curvature.

3.2 Curvature-Aware Updates without the Hessian

In this section, we derive Caffeine's Hessian-free, curvature-aware gradient ascent update. Section 3.2.1 shows how Caffeine replaces the Hessian with a single scalar. Section 3.2.2 shows how to obtain the scalar from two gradients instead of the Hessian matrix. Section 3.2.3 combines the per-checkpoint ascent steps along the training trajectory into the unlearning update.

3.2.1 Reducing the Hessian to a Single Scalar

We now illustrate how Caffeine calibrates the step size of the forget set without inverting the Hessian. Let $H_f(\theta) = \nabla^2_\theta L(D_f, \theta)$ denote the Hessian of the loss restricted to the forget set D_f, as opposed to H_{θ^*} (Equation (2)), which averages curvature over the full training set. Because Caffeine never has access to the retain set (Section 2), only $H_f(\theta)$ (not H_{θ^*}) is available to it (Appendix C.1). $H_f(\theta)$ can differ substantially from H_{θ^*}, especially for class deletion, where D_f and D_{ret} follow different distributions (Appendix C.1); Section 4 compares the two choices.

Rather than use this matrix directly, Caffeine approximates it by $K\mathbf{I}$, where $\mathbf{I}$ is the identity matrix and K is a scalar, so that the inverse Hessian in Equation (3) is replaced by $\frac{1}{K}\mathbf{I}$ and the $O(N^3)$ inversion reduces to a scalar division. A large K indicates sharp curvature (a short step), whereas a small K indicates flat curvature (a long step). To define K, we measure the curvature specifically in the direction of the unlearning objective: the forget-set gradient $g_f(\theta) = \nabla_\theta L(D_f, \theta)$. The curvature of the loss along $g_f(\theta)$, given by the Rayleigh quotient of the symmetric Hessian matrix $H_f(\theta)$, is:

$$K = \frac{g_f(\theta)^\top H_f(\theta) g_f(\theta)}{\|g_f(\theta)\|_2^2} \tag{5}$$

By the Courant-Fischer Min-Max Theorem, this Rayleigh quotient lies exactly between the smallest and largest eigenvalues of $H_f(\theta)$ (Proposition 1, supplementary material, proves $1/K$ approximates the local Newton step). Computing $H_f(\theta)g_f(\theta)$ directly, however, still requires second-order differentiation. In the next section, we show how to estimate K without computing $H_f(\theta)$.

3.2.2 A Finite-Difference Estimate of Curvature

Caffeine avoids computing $H_f(\theta)$ by estimating curvature from two gradients instead. The first gradient is $g_f(\theta)$, computed via one backpropagation pass over D_f. Moving along $+g_f(\theta)$ increases the loss on D_f, which is the direction Caffeine moves to remove D_f's influence.

The second gradient, the *perturbed gradient*, is evaluated after a small step $\Delta\theta$. A first-order Taylor expansion relates it to the first gradient as:

$$g_f(\theta + \Delta\theta) \approx g_f(\theta) + H_f(\theta)\Delta\theta \tag{6}$$

Because the Rayleigh quotient (Equation (5)) is identical for both $g_f(\theta)$ and $-g_f(\theta)$, Caffeine takes this probe step along $-g_f(\theta)$, while the unlearning update itself moves along $+g_f(\theta)$. We set the step size to the learning rate η that training used at that checkpoint, taking a probe step $\Delta\theta = -\eta \cdot g_f(\theta)$ along the negative gradient. Note that the expansion in Equation (6) holds only while $\eta\|g_f(\theta)\|$ is small, which is least likely at early checkpoints where gradients are largest (error bound in Proposition 2, Supplementary Material). Substituting this perturbation into Equation (6) allows us to approximate the exact Hessian-vector product:

$$g_f(\theta - \eta \cdot g_f(\theta)) \approx g_f(\theta) - \eta \cdot H_f(\theta) g_f(\theta) \quad \Longrightarrow \quad H_f(\theta) g_f(\theta) \approx \frac{g_f(\theta) - g_f(\theta - \eta \cdot g_f(\theta))}{\eta} \tag{7}$$

Substituting Equation (7) back into our Rayleigh quotient definition (Equation (5)), taking the absolute value to project negative curvature onto a positive-definite equivalent in non-convex regions, and adding a small stabilization constant c to the denominator, yields a proxy that approximates the Rayleigh quotient (which lies exactly between the smallest and largest eigenvalues) completely free of the Hessian:

$$K \approx \left| \frac{g_f(\theta)^\top \big(g_f(\theta) - g_f(\theta - \eta \cdot g_f(\theta))\big)}{\eta \|g_f(\theta)\|_2^2 + c} \right| \tag{8}$$

The only design choice in K is therefore the data on which these gradients are computed; we compare the forget set with the full training set in Section 4 and Appendix Section F.1. The constant $c > 0$ strictly prevents division by zero in regions where the loss landscape is perfectly flat. Because this relies on a global vector dot product, it ensures no single near-zero coordinate can dominate the estimate. Hence, K is computed from two gradients, the original gradient $g_f(\theta)$ and the perturbed gradient $g_f(\theta - \eta \cdot g_f(\theta))$. In the next section, we use K to estimate the influence of the forget set at each checkpoint.

Algorithm 1 Caffeine Unlearning

Input: θ^*, checkpoints Θ with rates η_t, forget set D_f, constants c, ϵ.
Output: $\theta_{unlearned}$.

1: $I_{sum}, I_{sq_sum} \leftarrow \mathbf{0}, \mathbf{0}$; $W_{total} \leftarrow 0$
2: **for** each checkpoint $\theta_t \in \Theta$ with rate η_t **do**
3: $g_f(\theta_t) \leftarrow \nabla_\theta L(D_f, \theta_t)$; $\theta_{temp} \leftarrow \theta_t - \eta_t g_f(\theta_t)$
4: $g_f(\theta_{temp}) \leftarrow \nabla_\theta L(D_f, \theta_{temp})$
5: $K_t \leftarrow \left| \frac{g_f(\theta_t)^\top (g_f(\theta_t) - g_f(\theta_{temp}))}{\eta_t \|g_f(\theta_t)\|_2^2 + c} \right|$; $I_t \leftarrow \frac{g_f(\theta_t)}{K_t + \epsilon}$
6: $I_{sum} \leftarrow I_{sum} + \eta_t I_t$; $I_{sq_sum} \leftarrow I_{sq_sum} + \eta_t I_t^2$; $W_{total} \leftarrow W_{total} + \eta_t$
7: $I_{total} \leftarrow Sign(I_{sum}) \odot \sqrt{I_{sq_sum}/W_{total}}$
8: **return** $\theta^* + I_{total}$

3.2.3 Curvature-Aware Ascent and Model Update

Let t index the saved checkpoints $\theta_t \in \Theta$ along the model's training trajectory. At each θ_t, the gradient $g_f(\theta_t)$ and curvature proxy K_t (Equation (8)) are combined into a localized ascent step I_t as

$$I_t = \frac{g_f(\theta_t)}{K_t + \epsilon} \tag{9}$$

where $\epsilon > 0$ is a small constant introduced for numerical stability to prevent division by zero in regions where the loss landscape is perfectly flat. Under the isotropic approximation, I_t estimates $H_f(\theta_t)^{-1} g_f(\theta_t)$. Because $\mathcal{I} = -H^{-1}\nabla_\theta L$, Equation (4) moves θ^* along $+H^{-1}\nabla_\theta L$, so Caffeine adds I_t rather than subtracting it; unlike Equation (4), I_t is not scaled by $|D_f|/n$. Caffeine then aggregates the per-checkpoint influences using three trajectory statistics, each weighted by the checkpoint's learning rate η_t to reflect the size of the training step at that checkpoint under SGD:[1]

$$I_{sum} = \sum_{t \in \Theta} \eta_t I_t \qquad I_{sq_sum} = \sum_{t \in \Theta} \eta_t I_t^2 \qquad W_{total} = \sum_{t \in \Theta} \eta_t \tag{10}$$

For each parameter, I_{sum} indicates whether the checkpoints' ascent steps share an overall direction or cancel out (sum to 0). I_{sq_sum} captures the size of the steps, and W_{total} normalizes I_{sq_sum}. The final update takes its sign from I_{sum} and its magnitude from the learning-rate-weighted root mean square (RMS) of the steps, and is added to the trained model to perform the deletion in a single step:

$$I_{total} = Sign(I_{sum}) \odot \sqrt{\frac{I_{sq_sum}}{W_{total}}}, \qquad \theta_{unlearned} = \theta^* + I_{total}, \tag{11}$$

where the square in Equation (10), and $Sign(\cdot)$, the product, and the square root in Equation (11), are applied element-wise. This time-weighted RMS acts as a dynamic trust region. It preserves the parameter's historical variance magnitude, ensuring a safely bounded step even in highly oscillatory regions of the trajectory (Remark 3 in Supplementary Material).

Caffeine. Algorithm 1 shows the complete working of Caffeine. It takes the model trained on the complete dataset, θ^*, the saved checkpoints Θ, their learning rates η_t, and the forget set D_f, and returns $\theta_{unlearned}$. Line 1 initializes the three running statistics to zero. Within the loop (lines 2–6), Caffeine visits each saved checkpoint once: it computes the forget-set gradient and takes one step of size η_t to a temporary point θ_{temp} (line 3), recomputes the gradient there (line 4), evaluates the curvature scalar K_t and the influence I_t (line 5, Equations (8) and (9)), and updates the running sums (line 6, Equation (10)). Once every checkpoint is visited, line 7 computes the update I_{total} via the learning-rate-weighted RMS, and line 8 adds it to θ^* (Equation (11)).

Complexity Analysis: **Space:** Caffeine keeps a constant number of model-sized vectors (the checkpoint, two gradients, and the running sums) plus one batch of activations, so its working memory is $O(N)$ independent of C; the C stored checkpoints add $O(C \cdot N)$ storage. **Time:** Caffeine uses only forget-set gradients, so each checkpoint costs $O(|D_f| \cdot N)$ and all C checkpoints $O(C \cdot |D_f| \cdot N)$.

[1] TracIn [29] similarly weights each checkpoint by its learning rate.

4 EVALUATION

Our evaluation answers four questions: (1) Does Caffeine remove one or more classes under class deletion? (2) Does it remove samples under random-subset deletion as retraining does? (3) What are its latency and memory costs? and (4) How sensitive is it to its operating parameters?

Experimental Setup. We run all experiments on NVIDIA A100 PCIe GPUs (80 GB) and report the mean over three runs (seeds 42, 1024, and 5). Each model is trained with SGD under a cosine annealing learning-rate schedule from an initial rate of $\eta = 0.01$, for 50 epochs at batch size 32, saving $C = 10$ checkpoints (one every 5 epochs). We set $c = 10^{-8}$ and $\epsilon = 0.01$.[2]

Benchmarking Datasets. We evaluate on five datasets spanning images and text data.

Image datasets. Each dataset has at least ten classes where the deletion request entails forgetting entire image class. All image experiments use **ResNet-18** [16] ($\sim$11.7M parameters), with a final layer sized to each dataset: **CIFAR-10** (10 classes) [7], **TinyImageNet** (200 classes) [25], and a **Kaggle face-recognition** dataset (17 classes) [36] of real celebrities.

Text datasets. A deletion request for text dataset entails forgetting a user's online comments: **IMDB**, five sentiment classes from "very negative" to "very positive" [35], and **Amazon Reviews**, five classes for one- to five-star ratings [45]. We fine-tune **DistilBERT** [31] (`distilbert-base-uncased`, $\sim$66.3M parameters) with a linear classification head over the five output classes.

Metrics. We evaluate Caffeine and comparable models along four axes: (1) *Retain and Test Accuracy*: accuracy on the retained training data and on unseen test data, (2) *Forget Accuracy*: accuracy on D_f. For *class deletion*, the target is 0%. For *random-subset deletion*, D_f follows the test distribution, so the target is test accuracy, and we report the gap $|\text{Acc}_{\text{forget}} - \text{Acc}_{\text{test}}|$. (3) *MIA AUC*: we run membership inference [34] with the loss threshold attack of [43], distinguishing D_f from an equal-sized test set; because an AUC below 0.5 also separates the two sets, we compare the $|\text{AUC} - 0.5|$ of each method with the retraining. This is implemented only for random-subset deletion requests, as MIA after class deletion would be similar to what forget accuracy already shows. (4) *Time and Memory*: runtime and peak GPU memory per deletion request.

Baselines. We test against *four* state-of-the-art approximate unlearning methods, described below. Only SFMU shares our source-free constraint; HFOL, GradTao and SCRUB read the retain set and so have strictly more information than Caffeine. HFOL stands in for influence-function methods [20]; we do not run generic inverse-HVP estimators such as LiSSA [1] or conjugate gradient [27] as unlearning updates. All methods, including retraining, start from the same original models and checkpoints, so we compare each against retraining on the same model. Each baseline uses the authors' released code and recommended hyperparameters (settings and reproduction details in Appendix D).

HFOL [30] precomputes the influence (using a vector) for every sample in the training dataset. Such precomputation for the entire dataset makes it slow and memory-heavy (4.70–$14.74\times$ more than Caffeine, and 1 vector/sample).

SFMU [2] unlearns the forget set from the model's final layer, which does not sufficiently represent the feature space but SFMU has compute and memory requirements comparable to Caffeine.

GradTao [38] uses a small validation set which mimics the forget set. Although this allows memory and time efficient computations, it is not an accurate representation of the forget set.

SCRUB [22] fine-tunes a student copy of the model on retain set while disobeying it on forget set. Since the two sets share features, this reduces the performance of the retain set and also multiple copies makes it expensive in both runtime and memory.

4.1 CLASS DELETION ON IMAGE DATASETS

CIFAR-10. We gradually grow the deletion for the CIFAR-10 dataset (Table 1): first one class (class 0, 10% of the data), then two (classes 0–1, 20%), and finally three (classes 0–2, 30%). The test

[2] Code: https://anonymous.4open.science/r/Caffeine-0770

Table 1: Class deletion on CIFAR-10; sequentially unlearning up to 3 classes (10%–30% dataset).

Method	1 Class (C0)					2 Classes (C0–1)					3 Classes (C0–2)				
	Test (%)	Forget (%)	Retain (%)	Time (s)	Memory (MB)	Test (%)	Forget (%)	Retain (%)	Time (s)	Memory (MB)	Test (%)	Forget (%)	Retain (%)	Time (s)	Memory (MB)
Original	86.58	88.62	88.84	1047	1178.43	86.58	88.62	88.84	1047	1178.43	86.58	88.62	88.84	1047	1178.43
Retrain	85.48	0.00	89.98	942	1557.48	75.08	0.00	88.75	837	1107.53	65.78	0.00	88.78	733	774.33
SFMU	82.55	89.96	88.35	26	973.76	82.22	88.61	87.87	52	1430.82	81.08	85.16	87.97	79	1692.19
HFOL	10.33	100.00	0.00	5807	9142.25	Failed*	Failed	Failed	Failed	Failed	Failed*	Failed	Failed	Failed	Failed
GradTao	51.3	0.0	57.8	26	1753.85	50.4	0.0	64.2	47	1710.05	31.3	0.0	43.3	69	1708.80
SCRUB	38.7	0.0	42.8	492	1369.60	24.1	0.0	30.6	451	1372.35	16.9	0.0	23.8	407	1462.40
Caffeine_forget	**81.13**	**0.04**	**87.86**	**41**	**961.5**	**73.25**	**0.54**	**85.49**	**51**	**1040.91**	**62.62**	**1.37**	**82.39**	**57**	**1517.14**
Caffeine_full	**83.49**	**0.01**	**88.76**	**472**	**1204.5**	**74.84**	**0.22**	**86.11**	**469**	**1225.67**	**64.38**	**0.75**	**83.49**	**481**	**1215.27**

* *Failed due to Out of Memory (OOM)*

Table 2: Class deletion on TinyImageNet dataset; we forget 20 classes.

Method	Test (%)	Forget (%)	Retain (%)	Time (s)	Memory (MB)
Original	51.08	62.43	60.65	4513	1546.33
Retrain	47.00	0.00	62.14	4103	1486.59
SFMU	46.08	12.59	58.37	162	1229.63
HFOL	51.17	61.94	60.68	3922	4818.61
GradTao	29.35	0.00	31.58	97	1395.22
SCRUB	32.65	6.07	34.71	1893	1352.76
Caffeine	**46.37**	**0.00**	**59.07**	**170**	**987.49**

Table 3: Class deletion on Kaggle face-recognition dataset; we forget one identity.

Method	Test (%)	Forget (%)	Retain (%)	Time (s)	Memory (MB)
Original	79.77	83.35	83.27	275	1451.11
Retrain	77.54	0.00	82.67	248	1306.57
SFMU	78.58	83.08	83.15	5	1079.26
HFOL	11.11	83.21	0.00	112	11641.77
GradTao	45.44	0.00	54.09	15	2103.61
SCRUB	34.67	0.00	39.54	93	1581.32
Caffeine	**73.87**	**0.11**	**79.46**	**12**	**789.70**

set includes the forgotten classes, so each deleted class puts a further 10% of it beyond the reach of any unlearned model, and test accuracy is expected to fall with every deletion even under retraining. Caffeine's test accuracy stays within 4.35, 1.83, and 3.16 points of retraining at 10%, 20%, and 30% deletion respectively, while its forget accuracy rises only marginally with the forget set, to 0.04%, 0.54%, and 1.37%, against retraining's 0%. At 10% deletion, Caffeine is 23× faster than retraining and uses 1.62× less memory; at 20%, 16.4× faster with 1.06× less memory; at 30%, still 12.9× faster but with 1.96× more memory. On CIFAR-10, HFOL runs out of memory when two or more classes are deleted, and in its one completed run it predicts a single class (0% retain accuracy). SFMU leaves forget accuracy near the original model's at every deletion level.

TinyImageNet. For TinyImageNet (Table 2), the 20 forgotten classes make up 10% of the test set, so part of the test-accuracy drop is expected even under retraining. Retraining's test accuracy drops by 4.08 points and Caffeine's by 4.71 points, while Caffeine reaches 0% forget accuracy. HFOL leaves forget accuracy nearly unchanged, while GradTao and SCRUB lose over 14 points of test accuracy relative to retraining. SFMU is the only other method that both keeps test accuracy near retraining's and substantially reduces forget accuracy (from 62.43% to 12.59%), but its forget accuracy remains 12.59 points above Caffeine's.

Face recognition. When forgetting one identity (Table 3), Caffeine reaches 0.11% forget accuracy and stays within 3.67 and 3.21 points of retraining's test and retain accuracy, while running 20.7× faster and using 1.65× less memory. SFMU leaves forget accuracy near the original model's; GradTao and SCRUB reach 0% forget accuracy but lose over 32 points of test accuracy; and HFOL's retain accuracy falls to 0%.

Should the curvature be determined from the forget set or the complete dataset. To answer this, we compare Caffeine with Caffeine_full, an ablation that computes K_t on the full training set D instead of D_f in Algorithm 1, so the gap between the two is the cost of the source-free constraint. On CIFAR-10 (Table 1) that cost is 2.4, 1.6, and 1.8 points of test accuracy at 10%, 20%, and 30% deletion, with forget accuracy near 0 for both. The price of measuring curvature on D is runtime: Caffeine_full is 8.4–11.5× slower, and its runtime grows with $|D_{ret}|$. The forget set is therefore the better choice, and we use Caffeine throughout (further discussion in Appendix F).

4.2 Class Deletion on Text Datasets

IMDB. On IMDB, the deletion request targets the 'very negative' class. The original model fits its training data perfectly (100% retain accuracy) but reaches only 50.15% test accuracy, so it overfits strongly. Among the approximate methods, Caffeine is the only one that brings forget accuracy near 0% (0.92%) while keeping test accuracy within 2.56 points of retraining's. GradTao and SCRUB also reach 0% forget accuracy, but lose over 18 points of test accuracy, and GradTao uses 1.9× more memory than Caffeine. Caffeine's retain accuracy, however, is 15.11 points below retraining's; it performs the deletion 2.3× faster than retraining with 1.1× less memory.

Table 4: Class deletion on IMDB dataset; we forget the 'very negative' class.

Method	Test (%)	Forget (%)	Retain (%)	Time (s)	Memory (MB)
Original Model	50.15	100.00	100.00	402	7574.56
Retrain	47.36	0.00	100.00	381	7171.31
SFMU	50.08	100.00	100.00	116	6628.26
HFOL	Failed*	Failed	Failed	Failed	Failed
GradTao	23.38	0.00	33.55	185	12533.50
SCRUB	28.72	0.00	34.11	2119	9852.04
Caffeine	**44.80**	**0.92**	**84.89**	**167**	**6627.03**

* *Failed due to lack of PyTorch backward derivative*

Table 5: Class deletion on Amazon Reviews dataset; we forget the '1 star' class.

Method	Test (%)	Forget (%)	Retain (%)	Time (s)	Memory (MB)
Original Model	55.87	90.68	86.58	565	7277.45
Retrain	46.73	0.00	89.70	471	7183.62
SFMU	48.13	89.11	88.20	258	5363.99
HFOL	Failed*	Failed	Failed	Failed	Failed
GradTao	19.77	0.00	25.00	239	11525.91
SCRUB	19.47	100.00	0.00	2646	8588.06
Caffeine	**41.40**	**1.22**	**85.58**	**219**	**6271.43**

* *Failed due to lack of PyTorch backward derivative*

Table 6: Random-Subset deletion on CIFAR-10 dataset; we forget a random 10% subset.

Method	Test (%)	Forget (%)	Retain (%)	MIA	Time (s)	Memory (MB)
Original	84.58	88.62	88.84	0.52	1047	1178.43
Retrain	85.02	85.16	89.81	0.48	952	1150.05
SFMU	79.30	83.20	83.18	0.54	292	1062.27
HFOL	10.30	10.18	10.24	0.53	3973	9232.73
GradTao	59.82	58.35	61.56	0.46	27	1745.75
SCRUB	67.4	66.36	68.91	0.46	916	1411.92
Caffeine	**82.35**	**83.70**	**86.25**	**0.4760**	**49.47**	**927.13**
Caffeine_full	**84.44**	**85.21**	**87.88**	**0.4860**	**534**	**1287.11**

Table 7: Random-Subset deletion on TinyImageNet dataset; we forget a random 10% subset.

Method	Test (%)	Forget (%)	Retain (%)	MIA	Time (s)	Memory (MB)
Original	51.08	61.22	60.54	0.52	4617	1583.28
Retrain	50.58	47.41	61.17	0.49	4198.56	1447.68
SFMU	50.03	58.49	59.72	0.54	163	1012.24
HFOL	51.15	61.36	61.03	0.55	3968	4636.21
GradTao	31.03	29.51	30.91	0.37	99	1550.97
SCRUB	33.48	30.54	33.52	0.38	1930	1453.97
Caffeine	**50.12**	**45.27**	**58.21**	**0.47**	**166**	**987.26**
Caffeine_full	Failed*	Failed	Failed	Failed	Failed	Failed

* *Failed due to Out of Memory (OOM)*

Amazon Reviews. On Amazon Reviews, the deletion request targets the '1 star' class. GradTao reaches 0% forget accuracy but only 19.77% test accuracy, whereas Caffeine keeps forget accuracy at 1.22%, with test and retain accuracy 5.33 and 4.12 points below retraining's. SCRUB collapses to predicting the forgotten class (0% retain, 100% forget accuracy).

4.3 Random-Subset Deletion

CIFAR-10. We delete a random 10% of CIFAR-10 (Table 6). Retraining is the reference model, with a test–forget gap of 0.14 points and an MIA AUC of 0.48 (0.011 from chance); the original model's AUC of 0.50 shows that this attack barely separates members even before unlearning. Caffeine trails retraining by 2.67 and 1.46 points in test and forget accuracy, with a test–forget gap of 1.35 points; its MIA AUC of 0.47 (0.024 from chance) separates the forget set more than retraining does but less than every baseline (0.034–0.041). GradTao and SCRUB lose over 17 points of test accuracy relative to retraining, and HFOL collapses to about 10% accuracy on every set, at 4.2× retraining's runtime and 8.0× its memory. SFMU has a large test–forget gap (3.90 points).

TinyImageNet. We delete a random 10% of TinyImageNet's training data (Table 7). Retraining's forget accuracy is 3.17 points below its test accuracy. HFOL uses 3.2× more memory than retraining. Caffeine's test–forget gap is 4.85 points, its forget accuracy is within 2.14 points of retraining's, and its MIA AUC of 0.47 is 0.03 from chance, against 0.01 for retraining and 0.02 for the original model; it requires 25× less time and 1.4× less memory.

4.4 Ablation Study

We study Caffeine's sensitivity to three hyperparameters: (1) training learning rate, (2) number of stored checkpoints, and (3) training batch size. The defaults values are $\eta = 0.01$, $C = 10$, batch size 32, and where the class 0 is deleted from CIFAR-10 (Figure 1).

Learning Rate. We vary the training learning rate η from 0.001 to 0.1 (Figure 1(a)). At $\eta = 0.001$, forget accuracy is already near 0% because the undertrained model has learned little about the forgotten class. We find that Caffeine strikes the best balance at $\eta = 0.01$, which is within 2.9%, 2.5%, and 2.7% of retraining's forget, retain, and test accuracy, therefore we use $\eta = 0.01$ throughout.

Trajectory Checkpoints. We vary the number of stored checkpoints C from 1 to 10 over a fixed 50-epoch training run (Figure 1(b)). With only 1 or 2 checkpoints, the trajectory is sampled too coarsely and mostly misses how the forget set shaped training (accuracies fall well below retraining). At $C = 5$ the gap narrows, and at $C = 10$ (our selected checkpoints), Caffeine lands within 4.5%, 6.0%, and 4.6% of retraining's test, forget, and retain accuracy.

Batch Size. We vary the training batch size from 8 to 128 (Figure 1(c)). A batch size of 32 works best as Caffeine lands within 4.9% and 0.9% of retraining's test and forget accuracy. Both smaller (8–16)

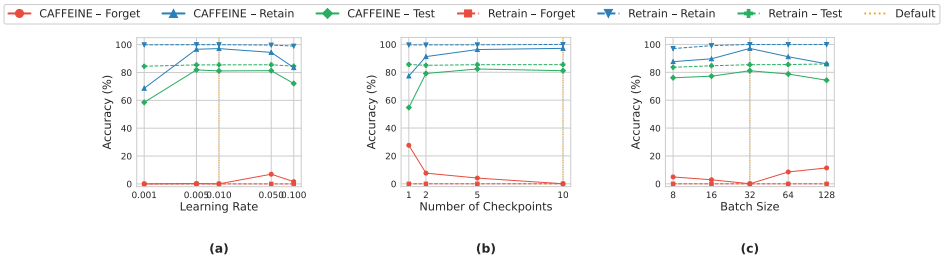

Figure 1: Ablation study on CIFAR-10 (single-class deletion). Dashed lines represent retraining.

and larger (64–128) batches move Caffeine further from retraining. Because Caffeine computes g_f over all of D_f, the training batch size affects it only through the trained model and its checkpoints.

A more detailed discussion of Ablation Study appears in Appendix F.

5 Related Work

Hessian-Based Unlearning. Influence functions [20] estimate a sample's effect on the trained model using the Hessian matrix H, at substantial computational cost, and are known to be fragile due to the non-convexity of deep networks [3]. Grosse et al. [13] scale influence functions to large language models by approximating each layer's curvature as a Kronecker product of two small matrices, reducing Hessian inversion to those matrices. Separately, other work avoids the full Hessian using low-rank structure or adapters [8, 17–19, 21, 24, 26, 44]. These methods, however, still use the retain set to estimate curvature, whereas Caffeine uses only the forget set, at an $O(N)$ update cost.

Hessian-Free Unlearning. Retraining on the retain set is the gold standard for exact unlearning; SISA [4] shards the data to reduce this cost, but a class-level deletion spanning all shards is similar to a full retrain. Certified unlearning [15] bounds the statistical distance between unlearned and retrained models theoretically; Caffeine instead validates this empirically via MIA AUC. Exact deletion also exists for trees and regression models [23, 32, 33, 41, 42], but does not extend to deep networks. DeltaGrad [40] similarly reuses stored gradients to accelerate retraining via an L-BFGS curvature estimate, but still needs the retain set; Amnesiac ML [12] reverses exact stored per-batch updates, viable only for few affected batches; and Unrolling SGD [37] approximates unlearning from only the initial weights, needing a training-time regularizer to stay accurate. In the source-free setting, Chundawat et al. [6] use noise or distillation but only at the class level. Fisher forgetting [11] approximates the Fisher information matrix as diagonal, weakening under class-deletion queries; SSD [10] dampens forget-specialized parameters via the same matrix, improving its cost, but still needs training-data statistics. STaR [48] suppresses outputs at inference rather than editing weights, and SalUn [9] masks weights by forget-set gradient magnitude, but still needs a full-network gradient pass and leaves its update uncalibrated. Caffeine is designed for the centralized setting, but it extends even to federated setting, where clients compute and transmit gradients during training and could estimate curvature locally [5, 14, 28, 39, 46, 47]. Appendix A includes a more detailed related work.

6 Discussion and Conclusion

Discussion. The stored checkpoints carry the forget set's influence even after a deletion request is served, and so must remain private in the same way D_f itself must. We assume the deployment setting of source-free unlearning: checkpoints stay on the training side, and only the unlearned model θ_u is exposed [2].

Conclusion. In this paper, we introduced Caffeine, an approximate machine unlearning algorithm that operates as a curvature-aware gradient ascent, keeping the calibrated step size of second-order methods without ever requiring the retained dataset or the global Hessian matrix. Caffeine estimates the scalar curvature proxy from two forget-set gradients via a finite-difference Taylor expansion, and aggregates these localized ascent steps via a time-weighted RMS to safely bound the final update spanning every layer of the model. Our tests across class and random-subset deletion queries on five image and text datasets illustrate that Caffeine matches retraining's erasure while using up to $1.65\times$ less memory and 2.2–$25.3\times$ less time.

USE OF LARGE LANGUAGE MODELS

We used generative AI tools to aid and polish the writing of this paper. All AI suggestions were reviewed by the authors, and all reported results come from our own experiments. We take responsibility for the final content of this work.

REFERENCES

[1] Naman Agarwal, Brian Bullins, and Elad Hazan. Second-order stochastic optimization for machine learning in linear time, 2017. URL https://arxiv.org/abs/1602.03943.

[2] Sk Miraj Ahmed, Umit Yigit Basaran, Dripta S. Raychaudhuri, Arindam Dutta, Rohit Kundu, Fahim Faisal Niloy, Basak Guler, and Amit K. Roy-Chowdhury. Towards source-free machine unlearning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[3] Samyadeep Basu, Philip Pope, and Soheil Feizi. Influence functions in deep learning are fragile. *arXiv preprint arXiv:2006.14651*, 2020.

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

[5] Tianshi Che, Yang Zhou, Zijie Zhang, Lingjuan Lyu, Ji Liu, Da Yan, Dejing Dou, and Jun Huan. Fast federated machine unlearning with nonlinear functional theory. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, Proceedings of Machine Learning Research, pp. 4241–4268. PMLR, 2023. URL https://proceedings.mlr.press/v202/che23b.html.

[6] Vikram S. Chundawat, Ayush K. Tarun, Murari Mandal, and Mohan Kankanhalli. Zero-shot machine unlearning. *IEEE Transactions on Information Forensics and Security*, 18:2345–2354, 2023.

[7] Will Cukierski. Cifar-10 - object recognition in images. https://kaggle.com/competitions/cifar-10, 2013. Kaggle.

[8] Chenlu Ding, Jiancan Wu, Yancheng Yuan, Jinda Lu, Kai Zhang, Alex Su, Xiang Wang, and Xiangnan He. Unified parameter-efficient unlearning for llms. In *International Conference on Learning Representations*, 2025.

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

[10] Jack Foster, Stefan Schoepf, and Alexandra Brintrup. Fast machine unlearning without retraining through selective synaptic dampening. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 12043–12051, 2024.

[11] 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 (CVPR)*, pp. 9301–9309, 2020.

[12] Laura Graves, Vineel Nagisetty, and Vijay Ganesh. Amnesiac machine learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 11516–11524, 2021.

[13] Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, Evan Hubinger, Kamilė Lukošiūtė, Karina Nguyen, Nicholas Joseph, Sam McCandlish, Jared Kaplan, and Samuel R. Bowman. Studying large language model generalization with influence functions. *CoRR*, abs/2308.03296, 2023. doi: 10.48550/ARXIV.2308.03296. URL https://doi.org/10.48550/arXiv.2308.03296.

[14] Hanlin Gu, Win Kent Ong, Chee Seng Chan, and Lixin Fan. Ferrari: Federated feature unlearning via optimizing feature sensitivity. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[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 (ICML)*, pp. 3832–3842, 2020.

[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 (CVPR)*, pp. 770–778, 2016.

[17] Zhehao Huang, Xinwen Cheng, JingHao Zheng, Haoran Wang, Zhengbao He, Tao Li, and Xiaolin Huang. Unified gradient-based machine unlearning with remain geometry enhancement. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/2e622ac74f66df03b686a12e2e0e4424-Abstract-Conference.html.

[18] Masaru Isonuma and Ivan Titov. Unlearning traces the influential training data of language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024.

[19] Jinghan Jia, Yihua Zhang, Yimeng Zhang, Jiancheng Liu, Bharat Runwal, James Diffenderfer, Bhavya Kailkhura, and Sijia Liu. SOUL: unlocking the power of second-order optimization for LLM unlearning. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pp. 4276–4292. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.EMNLP-MAIN.245. URL https://doi.org/10.18653/v1/2024.emnlp-main.245.

[20] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning (ICML)*, pp. 1885–1894. PMLR, 2017.

[21] Aleksey Kudelya and Alexander Shirnin. Lacuna inc. at semeval-2025 task 4: Lora-enhanced influence-based unlearning for llms. In *Proceedings of the 19th International Workshop on Semantic Evaluation (SemEval-2025)*, pp. 1528–1533, 2025.

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

[23] Meghdad Kurmanji, Eleni Triantafillou, and Peter Triantafillou. Machine unlearning in learned databases: An experimental analysis. In *Proceedings of the ACM on Management of Data (SIGMOD)*, volume 2. ACM, 2024.

[24] Yongchan Kwon, Eric Wu, Kevin Wu, and James Zou. Datainf: Efficiently estimating data influence in lora-tuned llms and diffusion models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=9m02ib92Wz.

[25] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

[26] Jiawei Liu, Chenwang Wu, Defu Lian, and Enhong Chen. Efficient machine unlearning via influence approximation. *arXiv preprint arXiv:2507.23257*, 2025.

[27] James Martens. Deep learning via hessian-free optimization. In Johannes Fürnkranz and Thorsten Joachims (eds.), *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pp. 735–742. Omnipress, 2010. URL https://icml.cc/Conferences/2010/papers/458.pdf.

[28] Chao Pan, Jin Sima, Saurav Prakash, Vishal Rana, and Olgica Milenkovic. Machine unlearning of federated clusters. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=VzwfoFyYDga.

[29] Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/e6385d39ec9394f2f3a354d9d2b88eec-Abstract.html.

[30] Xinbao Qiao, Meng Zhang, Ming Tang, and Ermin Wei. Hessian-free online certified unlearning. In *International Conference on Learning Representations (ICLR)*, 2025.

[31] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[32] Sebastian Schelter, Stefan Grafberger, and Ted Dunning. Hedgecut: Maintaining randomised trees for low-latency machine unlearning. In *Proceedings of the 2021 International Conference on Management of Data (SIGMOD)*, pp. 1545–1557. ACM, 2021.

[33] Sebastian Schelter, Stefan Grafberger, and Maarten de Rijke. Snapcase - regain control over your predictions with low-latency machine unlearning. *Proc. VLDB Endow.*, 17(12):4273–4276, 2024. doi: 10.14778/3685800.3685853. URL https://www.vldb.org/pvldb/vol17/p4273-schelter.pdf.

[34] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pp. 3–18. IEEE Computer Society, 2017. doi: 10.1109/SP.2017.41. URL https://doi.org/10.1109/SP.2017.41.

[35] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard (eds.), *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://aclanthology.org/D13-1170/.

[36] Vishesh Thakur. Celebrity face image dataset. https://www.kaggle.com/datasets/vishesh1412/celebrity-face-image-dataset, 2022.

[37] Anvith Thudi, Gabriel Deza, Varun Chandrasekaran, and Nicolas Papernot. Unrolling sgd: Understanding factors influencing machine unlearning. In *2022 IEEE 7th European symposium on security and privacy (EuroS&P)*, pp. 303–319. IEEE, 2022.

[38] Daniel Trippa, Cesare Campagnano, Maria Sofia Bucarelli, Gabriele Tolomei, and Fabrizio Silvestri. Gradient-based and task-agnostic machine unlearning. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, pp. XXXXX–YYYYY. PMLR, 2024.

[39] Chen Wu, Sencun Zhu, and Prasenjit Mitra. Federated unlearning with knowledge distillation. *arXiv preprint arXiv:2201.09441*, 2022.

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

[41] Yinjun Wu, Val Tannen, and Susan B. Davidson. Priu: A provenance-based approach for incrementally updating regression models. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pp. 1727–1741. ACM, 2020.

[42] Zhaomin Wu, Junhui Zhu, Qinbin Li, and Bingsheng He. Deltaboost: Gradient boosting decision trees with efficient machine unlearning. In *Proceedings of the ACM on Management of Data (SIGMOD)*, volume 1. ACM, 2023.

[43] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *31st IEEE Computer Security Foundations Symposium, CSF 2018, Oxford, United Kingdom, July 9-12, 2018*, pp. 268–282. IEEE Computer Society, 2018. doi: 10.1109/CSF.2018.00027. URL https://doi.org/10.1109/CSF.2018.00027.

[44] Binchi Zhang, Yushun Dong, Tianhao Wang, and Jundong Li. Towards certified unlearning for deep neural networks. *arXiv preprint arXiv:2408.00920*, 2024.

[45] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

[46] Yian Zhao, Pengfei Wang, Heng Qi, Jianguo Huang, Zongzheng Wei, and Qiang Zhang. Federated unlearning with momentum degradation. *IEEE Internet of Things Journal*, 11: 8860–8870, 2024.

[47] Zhengyi Zhong, Weidong Bao, Ji Wang, Shuai Zhang, Jingxuan Zhou, Lingjuan Lyu, and Wei Yang Bryan Lim. Unlearning through knowledge overwriting: Reversible federated unlearning via selective sparse adapter. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 30661–30670, 2025.

[48] Jingjing Zhou, Gaoxiang Cong, Li Su, and Liang Li. Star: Sensitive trajectory regulation for unlearning in large reasoning models. In Sven Koenig, Chad Jenkins, and Matthew E. Taylor (eds.), *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pp. 35121–35129. AAAI Press, 2026. doi: 10.1609/AAAI.V40I41.40818. URL https://doi.org/10.1609/aaai.v40i41.40818.