
The Unseen Threat: Residual Knowledge in Machine Unlearning under Perturbed Samples

Anonymous Author(s)

Abstract

Machine unlearning offers a practical alternative to avoid full model re-training by approximately removing the influence of specific user data. While existing methods certify unlearning via statistical indistinguishability from re-trained models, these guarantees do not naturally extend to model outputs when inputs are adversarially perturbed. In particular, slight perturbations of forget samples may still be correctly recognized by the unlearned model—even when a re-trained model fails to do so—revealing a novel privacy risk: information about the forget samples may persist in their local neighborhood. In this work, we formalize this vulnerability as residual knowledge and show that it is inevitable in high-dimensional settings. To mitigate this risk, we propose a fine-tuning strategy, named RURK, that penalizes the model’s ability to re-recognize perturbed forget samples. Experiments on vision benchmarks with deep neural networks demonstrate that residual knowledge is prevalent across existing unlearning methods and that our approach effectively prevents residual knowledge.

1 Introduction

The widespread use of user data in training machine learning (ML) models has raised significant privacy concerns, particularly when user data is memorized by models such as deep neural networks, and can later be extracted or reconstructed (Carlini et al., 2023; Li et al., 2024). This memorization violates regulations such as the “Right to be Forgotten” in EU’s GDPR (Voigt and Von dem Bussche, 2017), which mandates that, upon a user’s removal request, an ML service provider must not only delete the user data from databases but also ensure its removal from the ML models themselves (Shastri et al., 2019). As a result, simply deleting the data is often insufficient; instead, re-training the model from scratch without the specific user data is necessary. However, this re-training process is computationally expensive and impractical in real-time or large-scale settings (Ginart et al., 2019).

To address this challenge, *machine unlearning* has been proposed as a more scalable alternative that *approximately* removes the influence of the specific data (referred to as forget samples) from a pre-trained model, as a substitute for avoiding full re-training (Cao and Yang, 2015; Bourtole et al., 2021). This approach, known as *approximate unlearning*, is often certified through statistical indistinguishability between the unlearned model and one re-trained without the forget samples (Guo et al., 2020; Chourasia and Shah, 2023).

Theoretical formulations of approximate unlearning (cf. § 2) suggest that an unlearned model should behave similarly to a re-trained model on forget samples, for example, by producing similar predictions or classification accuracy. However, these guarantees typically apply only to the original samples and do not extend to their proximities, especially under imperceptible adversarial perturbations. In complex hypothesis spaces, even minor input changes can cause the unlearned and re-trained models to *disagree* in their predictions, creating an additional layer of privacy risk. A particularly concerning case is when a slightly perturbed forget sample is still correctly classified by the unlearned model but not by the re-trained one. This reveals the presence of *residual knowledge*—latent traces of the forget samples that still persist in the unlearned model (cf. Figure 1). Residual knowledge is a prevalent

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

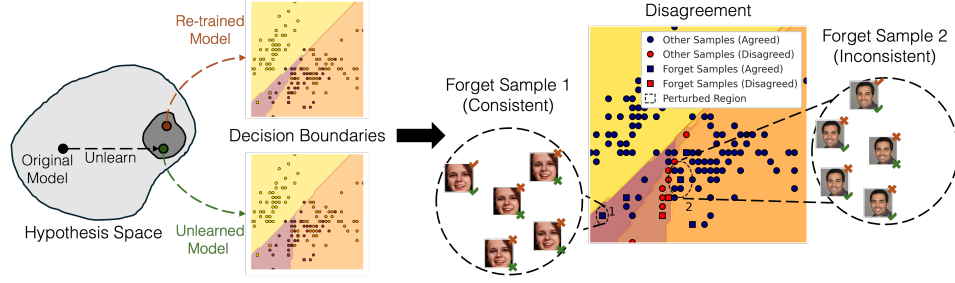


Figure 1: The re-trained (brown) and unlearned (green) models are statistically similar but may have slightly different decision boundaries (left), leading to disagreements on forget samples (right). Checkmarks and crosses on the images indicate correct and incorrect predictions from the re-trained model (top) and the unlearned model (bottom), respectively. Ideally—as shown with forget sample 1—both models should behave consistently across the original and all perturbed inputs. Residual knowledge in machine unlearning is illustrated by comparing prediction correctness: for forget sample 2, both models agree on the original sample, but the unlearned model correctly predicts more of its perturbed variants.

40 issue. For instance, on the CIFAR-10 dataset, when subjected to a small perturbation norm (≈ 0.03),
 41 over 7% of the forget samples still exhibit residual knowledge (cf. Appendix C).

42 In this paper, we study how adversarially perturbed inputs affect the indistinguishability between
 43 unlearned and re-trained models. In § 3, we show that adversarial examples can reliably distinguish
 44 between the two models, even under certified approximate unlearning. We formalize this observation
 45 by demonstrating that adversarial attacks can induce probabilistically distinguishable outputs. Further,
 46 using geometric probability (Talagrand, 1995), we prove that such disagreement is inevitable in high-
 47 dimensional input spaces. These findings underscore the need to explicitly incorporate robustness
 48 against such local disagreement into unlearning frameworks.

49 To capture this risk, § 4 introduces the notion of residual knowledge, which quantifies the likelihood
 50 that an unlearned model retains predictive traces of forget samples under perturbation. As a more
 51 tractable proxy for disagreement, residual knowledge exposes a vulnerability not addressed by current
 52 certification methods. To mitigate this, we propose RURK, a fine-tuning strategy for Robust Unlearning
 53 that suppresses Residual Knowledge while maintaining accuracy on the rest of the samples. Our
 54 approach identifies and penalizes perturbed inputs that the unlearned model still classifies correctly,
 55 thus enhancing robustness against residual knowledge. We empirically validate the existence of
 56 such local disagreement and residual knowledge across multiple unlearning algorithms in § 5. We
 57 further demonstrate that our fine-tuning strategy effectively reduces residual knowledge on standard
 58 vision datasets using deep neural networks. To the best of our knowledge, this is the first work to
 59 uncover this novel privacy risk and provide a scalable solution to address it. We conclude in § 6 with
 60 a discussion of limitations and future research directions.

61 Omitted proofs, additional explanations and discussions, details on experiment setups and training,
 62 and additional experiments are included in the Appendix. Code to reproduce our experiments will be
 63 made public soon.

64 2 Background and related work

65 Let $\mathcal{S} \triangleq \{\mathbf{s}_i = (\mathbf{x}_i, y_i)\}_{i=1}^n$ be a training dataset with $\mathbf{x}_i \in \mathcal{X} \subseteq \mathbb{R}^d$ and $y_i \in \mathcal{Y}$. The hypothesis
 66 space $\mathcal{H} \triangleq \{h_{\mathbf{w}} : \mathcal{X} \rightarrow \mathcal{Y}; \mathbf{w} \in \mathcal{W}\}$ consists of models parameterized by \mathbf{w} . We use h and \mathbf{w}
 67 interchangeably to refer to a model in \mathcal{H} . Let $\ell : \mathcal{W} \times \mathcal{S} \rightarrow \mathbb{R}^+$ be the loss function, and define
 68 the empirical risk as $L(\mathbf{w}, \mathcal{S}) = \frac{1}{n} \sum_{i=1}^n \ell(\mathbf{w}, \mathbf{s}_i)$. A (randomized) learning algorithm $A : \mathcal{S} \rightarrow \mathcal{H}$,
 69 such as Stochastic Gradient Descent (SGD), returns a model minimizing $L(\mathbf{w}, \mathcal{S})$ and induces a
 70 distribution over \mathcal{H} . We denote the ℓ_p -norm by $\|\mathbf{z}\|_p$, and the ℓ_p ball of radius τ centered at \mathbf{x} by
 71 $\mathcal{B}_p(\mathbf{x}, \tau) \triangleq \{\mathbf{z} \in \mathcal{X} : \|\mathbf{z} - \mathbf{x}\|_p \leq \tau\}$. Also, let $\mathbb{1}(\cdot)$ be the indicator function, $\text{surf}(\mathcal{Z})$ the surface
 72 area of a set \mathcal{Z} in \mathbb{R}^d , and $\mathcal{O}(\cdot)$ the big-O notation.

73 2.1 Machine unlearning

74 The forget set $\mathcal{S}_f \subseteq \mathcal{S}$ contains training samples to be removed. Machine unlearning is modeled as a
 75 randomized mechanism $M(A(\mathcal{S}), \mathcal{S}, \mathcal{S}_f)$ that removes the influence of \mathcal{S}_f from a pre-trained model

76 $A(\mathcal{S})$ (Xu et al., 2023). Like the learning algorithm itself, the output of the unlearning mechanism
 77 can be regarded as a random variable. A simple approach adds Gaussian noise to the model weights:
 78 $M(\mathbf{w}, \mathcal{S}, \mathcal{S}_f) = \mathbf{w} + \sigma \mathbf{n}$, with $\mathbf{n} \sim \mathcal{N}(0, \mathbf{I}_{|\mathbf{w}|})$ (Golatkhar et al., 2020a). However, as σ increases, the
 79 model becomes increasingly independent of the training data, potentially degrading performance. To
 80 preserve utility, unlearning should retain accuracy on the retain set $\mathcal{S}_r \triangleq \mathcal{S} \setminus \mathcal{S}_f$. A naïve yet effective
 81 strategy is to re-train the model from scratch on the retain set, i.e., $M(A(\mathcal{S}), \mathcal{S}, \mathcal{S}_f) = A(\mathcal{S}_r)$. While
 82 this achieves *exact unlearning*, it is often impractical due to the high computational cost of re-training
 83 for each unlearning request. Therefore, the core objective of machine unlearning is to efficiently
 84 eliminate the influence of \mathcal{S}_f while preserving utility on \mathcal{S}_r , all without full re-training.

85 This objective has spurred a vast literature on machine unlearning. Here, we outline the broad trends
 86 and defer details of specific algorithms to § 5. For comprehensive surveys, see Nguyen et al. (2022);
 87 Xu et al. (2023); Wang et al. (2024). Initial efforts focused on image classifiers, aiming to remove
 88 the influence of specific training images from a pre-trained model (Ginart et al., 2019; Wu et al.,
 89 2020; Neel et al., 2021; Sekhari et al., 2021; Izzo et al., 2021; Fan et al., 2023; Kurmanji et al., 2024;
 90 Goel et al., 2022; Zhang et al., 2024; Kodge et al., 2024). This idea was later extended to erasing
 91 abstract concepts (Ravfogel et al., 2022; Belrose et al., 2023), as well as adapting unlearning to
 92 broader model classes, including image generators (Li et al., 2024; Gandikota et al., 2023) and large
 93 language models (Eldan and Russinovich, 2023; Yao et al., 2024; Liu et al., 2025; Jang et al., 2022;
 94 Wang et al., 2023). Another line of work develops unlearning methods tailored to specific model
 95 families—such as linear classifiers (Guo et al., 2020), kernel methods (Golatkhar et al., 2020b; Zhang
 96 and Zhang, 2022), and tree-based models (Brophy and Lowd, 2021; Schelter et al., 2021).

97 2.2 Certification of machine unlearning

98 A valid unlearning mechanism ensures that the unlearned model $M(A(\mathcal{S}), \mathcal{S}, \mathcal{S}_f)$ is statistically
 99 similar to a re-trained model $A(\mathcal{S}_r)$. We denote the unlearned and re-trained models as random
 100 variables M and A , with corresponding distributions P_M and P_A , respectively. This requirement is
 101 formalized via (ϵ, δ) -indistinguishability:

102 **Definition 1** ((ϵ, δ) -indistinguishability). *Let X and Y be two random variables over a domain Ω .
 103 X and Y are said to be (ϵ, δ) -indistinguishable, also denoted as $X \stackrel{\epsilon, \delta}{\approx} Y$, if for all $\mathcal{T} \subseteq \Omega$*

$$e^{-\epsilon} (\Pr[Y \in \mathcal{T}] - \delta) \leq \Pr[X \in \mathcal{T}] \leq e^{\epsilon} \Pr[Y \in \mathcal{T}] + \delta. \quad (1)$$

104 This notion underlies differential privacy (DP) (Dwork and Roth, 2014) when X and Y are outputs on
 105 neighboring datasets. It also quantifies reproducibility of empirical findings when applied to models
 106 trained on independent samples from the same data distribution (Kalavasis et al., 2023; Impagliazzo
 107 et al., 2022; Bun et al., 2023).

108 Guo et al. (2020) were among the first to introduce certified machine unlearning using (ϵ, δ) -
 109 indistinguishability. An unlearning algorithm M satisfies (ϵ, δ) -unlearning if $M(A(\mathcal{S}), \mathcal{S}, \mathcal{S}_f) \stackrel{\epsilon, \delta}{\approx} A(\mathcal{S}_r)$;
 110 this reduces to exact unlearning when $\epsilon = \delta = 0$. Indistinguishability can be measured
 111 through various probability divergences as well. For instance, Chourasia and Shah (2023) and Chien
 112 et al. (2024) proposed (α, ϵ) -Rényi unlearning, which holds when the α -Rényi divergence¹ between
 113 P_M and P_A is bounded by ϵ , and can be translated into (ϵ, δ) -unlearning. Indeed, (α, ϵ) -Rényi
 114 unlearning can be converted to $(\epsilon + \log(1/\delta)/(\alpha - 1), \delta)$ -unlearning, for any $0 < \delta < 1$ (Mironov,
 115 2017). Both frameworks assume uniqueness or a unique stationary distribution of the empirical
 116 minimizer, which may limit their applicability in complex model classes.

117 Instead of comparing full model distributions P_M and P_A , a more practical certification framework
 118 evaluates unlearned models with readout functions $f : \mathcal{H} \times \mathcal{S} \rightarrow \mathbb{R}$ that an adversary might use to
 119 distinguish unlearned from re-trained models. Indistinguishability is then assessed via the output
 120 distribution of f , using divergences such as the Kullback-Leibler (KL) divergence². The certification
 121 condition requires $D_{\text{KL}}(\Pr[f(M, \mathcal{T})] \parallel \Pr[f(A, \mathcal{T})]) \leq \epsilon$ for any subset $\mathcal{T} \subseteq \mathcal{S}$, such as the forget,
 122 retain, or even hold-out test sets (Nguyen et al., 2020; Golatkhar et al., 2020a). This formulation
 123 flexibly captures different behaviors: f could represent a binary classifier (e.g., for Membership

¹For $\alpha > 1$, the α -Rényi divergence (Rényi, 1961) is defined as $D_\alpha(P|Q) \triangleq \frac{1}{\alpha-1} \log \mathbb{E}_Q \left[\left(\frac{P}{Q} \right)^\alpha \right]$.

²The KL divergence (Kullback and Leibler, 1951) is defined as $D_{\text{KL}}(P|Q) \triangleq \mathbb{E}_P[\log(P/Q)]$.

124 Inference Attack (MIA)) (Fan et al., 2023), utility metrics like accuracy, or re-learning time—the
 125 training epochs needed to re-learn S_f (Golatkhar et al., 2020a). Focusing on readout functions offers a
 126 more tractable and empirically grounded certification approach, especially for complex models.

127 3 Model indistinguishability and disagreement over sample perturbation

128 Statistical indistinguishability between unlearned and re-trained models can be certified theoretically—
 129 via (ϵ, δ) - or (α, ϵ) -Rényi unlearning—or empirically using a readout function (cf. §2). This, indeed,
 130 ensures similarity in model weights or outputs on forget/retain samples. However, such guarantees
 131 do not readily extend to perturbed inputs, even with imperceptible adversarial perturbations.

132 Several studies have examined the vulnerability of unlearning algorithms to adversarial attacks.
 133 Marchant et al. (2022) show that adversarial inputs can inflate the computational cost of unlearning,
 134 while Pawelczyk et al. (2025) find that poisoning attacks can prevent complete removal of forget
 135 samples. Zhao et al. (2024) further demonstrate that even a small number of malicious unlearning
 136 requests can degrade the adversarial robustness of the resulting model. However, these works
 137 primarily focus on the impact of adversarial perturbations on the unlearned models themselves.

138 This paper intends to explore a new dimension of how adversarial examples affect unlearning—
 139 specifically, how such examples may behave differently when fed to an unlearned model versus a
 140 re-trained one, even when the two satisfy statistical indistinguishability. Remarkably, despite this
 141 indistinguishability, it remains possible to craft adversarial inputs that distinguish between the two,
 142 revealing a novel privacy risk. This phenomenon, related to the transferability of adversarial examples
 143 (Tramèr et al., 2017), remains largely unexplored in the unlearning literature. The proofs of the
 144 propositions in this section are included in Appendix A.

145 3.1 Distinguishability of model output with adversarial examples

146 We begin by considering adversarial examples generated against either the unlearned or the re-trained
 147 model. The process of finding an adversarial example for a given input $\mathbf{x} \in \mathcal{X}$ can be formalized
 148 as a read-out function $g_{\mathbf{x}} : \mathcal{H} \rightarrow \mathcal{X}$, which may be deterministic or randomized. For instance, the
 149 minimum ℓ_2 -norm perturbation for a binary linear classifier, given by $g_{\mathbf{x}}(\mathbf{w}) = \mathbf{x} - (\mathbf{x}^\top \mathbf{x})\mathbf{w} / \|\mathbf{w}\|_2^2$,
 150 and the Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2016) are deterministic. In contrast,
 151 methods such as Projected Gradient Descent (PGD) (Madry et al., 2018) or random perturbations
 152 within $\mathcal{B}_p(\mathbf{x}, \tau)$ are randomized. Since generating adversarial examples can be seen as a post-
 153 processing of the model, the indistinguishability of adversarial examples can, in principle, be derived
 154 from the indistinguishability of the models themselves. Indeed, (ϵ, δ) -indistinguishability is preserved
 155 under arbitrary post-processing: if two random variables X and Y are (ϵ, δ) -indistinguishable, then
 156 so are $f(X)$ and $f(Y)$ for any (deterministic or randomized) function f . However, when considering
 157 adversarial examples of specific model realizations drawn from either the unlearning or re-training
 158 processes (cf. Lemma A.1), the level of indistinguishability can degrade, as formalized in the
 159 following proposition.

160 **Proposition 1** (Adversarial example on a model is less indistinguishable). *Suppose the unlearned*
 161 *$M(A(S), S, S_f)$ and re-trained $A(S_r)$ models are (ϵ, δ) -indistinguishable, and let \mathbf{x} be a fixed*
 162 *sample. Then with probability $2\delta/(1 - e^{-\epsilon})$, the adversarial example $g_{\mathbf{x}}(\cdot)$ found against the models*
 163 *$m \sim M$ or $a \sim A$ satisfies, for all $\mathcal{X}' \subseteq \mathcal{X}$*

$$\Pr[g_{\mathbf{x}}(m) \in \mathcal{X}'] \leq e^{-2\epsilon} \Pr[g_{\mathbf{x}}(a) \in \mathcal{X}'] \text{ or } e^{2\epsilon} \Pr[g_{\mathbf{x}}(a) \in \mathcal{X}'] \leq \Pr[g_{\mathbf{x}}(m) \in \mathcal{X}']. \quad (2)$$

164 Proposition 1 shows that even if the unlearned and re-trained models satisfy approximate unlearning
 165 certified by (ϵ, δ) -indistinguishability, the adversarial examples $g_{\mathbf{x}}(h)$ can become less indistinguish-
 166 able. Specifically, given a model h , with probability $2\delta/(1 - e^{-\epsilon})$, the distinguishability of the
 167 adversarial examples increases by a factor of two. Moreover, the indistinguishability assumption in
 168 Proposition 1 can be readily generalized to (α, ϵ) -Rényi unlearning.

169 3.2 Disagreement on adversarial examples is inevitable

170 In the previous section, Proposition 1 showed that even when the unlearned and re-trained models
 171 satisfy (ϵ, δ) -indistinguishability, there remains a nonzero probability that their likelihood ratio can

still be exploited to distinguish them via adversarial examples. However, evaluating the bound in Eq. (2) requires computing probabilities over the entire hypothesis space \mathcal{H} , which is often computationally intractable when \mathcal{H} is large or complex.

To address this challenge, we instead focus on model outputs at individual samples \mathbf{x} , which may belong to the forget or retain sets. We introduce a more tractable binary metric called *disagreement*, defined as $k(\mathbf{x}) = \mathbb{1}(M(\mathbf{x}) \neq A(\mathbf{x}))$, where $k(\mathbf{x}) = 1$ indicates that the unlearned and re-trained models produce different predictions at \mathbf{x} , and $k(\mathbf{x}) = 0$ otherwise. Disagreement has been widely used in the machine learning literature to study model behavior (Krishna et al., 2025; Uma et al., 2021), as well as prediction stability and bias (Kulynych et al., 2023). Notably, the (ϵ, δ) -indistinguishability guarantee over distributions can be translated into an upper bound on empirical disagreement across samples. In particular, when $\epsilon = \delta = 0$, perfect indistinguishability (i.e., exact unlearning) implies $k(\mathbf{x}) = 0$ for all $\mathbf{x} \in \mathcal{S}$. Ideally, we seek agreement not only on the training set but across the entire sample space \mathcal{X} , including unseen or perturbed data. Yet even small nonzero values of ϵ and δ can result in disagreement on certain inputs, particularly under adversarial perturbations, as we demonstrate in the following analysis.

To formalize this, we consider the sample space $\mathcal{X} = \mathbb{S}^{d-1} = \{\mathbf{x} \in \mathbb{R}^d \mid \|\mathbf{x}\|_2 = 1\}$, corresponding to the unit sphere in \mathbb{R}^d , where all data points are normalized to unit norm. In this setting, the disagreement function $k(\mathbf{x})$ can be viewed as a binary classifier over \mathbb{S}^{d-1} . We aim to bound the probability that disagreement occurs not only at a sample \mathbf{x} , but also within its local neighborhood under bounded perturbations, i.e., for $\mathbf{x}' \in \mathcal{B}_p(\mathbf{x}, \tau)$ such that $\|\mathbf{x} - \mathbf{x}'\|_p \leq \tau$. This motivates the need to understand how disagreement can propagate beyond the observed dataset to its local neighborhood in the broader sample space \mathcal{X} . A key mathematical tool for this purpose is the isoperimetric inequality (cf. Lemma A.2 and Talagrand (1995)). In probability theory and geometry, the isoperimetric inequality provides a lower bound on how the measure of a set expands when it is slightly “extended.” In high-dimensional spaces, particularly under uniform or Gaussian distributions, it formalizes the intuition that if a subset occupies a small volume, its boundary must be relatively large. In our context, this implies that even if two models disagree on only a small region, small perturbations can still cause disagreement to spread over a much larger region in the sample space. Combining this geometric insight with the definition of (ϵ, δ) -unlearning (cf. Definition 1), we establish the following result:

Proposition 2 (Inevitable disagreement). *Consider a sample $\mathbf{x} \in \mathbb{S}^{d-1}$. Let M and A denote the unlearned and re-trained models, respectively, satisfying (ϵ, δ) -unlearning. Suppose³ the agreement region satisfies $\text{surf}(\{\mathbf{x} \in \mathbb{S}^{d-1} \mid k(\mathbf{x}) = 0\}) / \text{surf}(\mathbb{S}^{d-1}) \leq 1/2$. Then with probability at least*

$$\left(\frac{2\delta}{1 - e^{-\epsilon}} \right) \left\{ 1 - \mathcal{O} \left[\left(\frac{\pi}{8} \right)^{1/2} \exp \left(-2\epsilon - \frac{d-1}{2} \tau^2 \right) \right] \right\}, \quad (3)$$

either $k(\mathbf{x}) = 1$, or there exist $\mathbf{x}' \in \mathcal{B}_p(\mathbf{x}, \tau)$ such that $k(\mathbf{x}') = 1$, where $p = 2$ or $p = \infty$.

This result shows that when $\epsilon = \delta = 0$, the probability in Eq. (3) is zero, since the unlearned and re-trained models are identical and cannot disagree. Conversely, as $\epsilon \rightarrow \infty$ for a fixed δ —reflecting maximal distinguishability between M and A —the probability of disagreement approaches its upper bound of 2δ . More generally, for any fixed (ϵ, δ) , the probability in Eq. (3) depends solely on the perturbation norm τ : as τ increases, the probability of disagreement rises, as adversarial examples explore a broader neighborhood around the input. Moreover, Proposition 2 holds for both $p = 2$ and $p = \infty$, aligning with the common practice of measuring perturbation size using either the ℓ_2 (Euclidean) norm or the ℓ_∞ (maximum) norm in prior works, e.g., Goodfellow et al. (2014); Madry et al. (2018).

4 Removing residual knowledge in unlearned models

Proposition 2 highlights that disagreement between the unlearned and re-trained models can persist even when an unlearning algorithm satisfies the (ϵ, δ) -unlearning constraint. In particular, although the two models may agree on the original input (\mathbf{x}, y) , it is often possible to craft an adversarial example with imperceptible perturbations (viz., small τ) that causes them to behave differently, thereby revealing a potential privacy risk.

³This condition can be easily satisfied in multi-class settings, i.e., $|\mathcal{Y}| > 2$.

We refer to such disagreement between specific unlearned and re-trained models m and a as *adversarial disagreement*⁴, defined as $k_\tau(\mathbf{x}') = \mathbb{1}(m(\mathbf{x}') \neq a(\mathbf{x}'))$ for $\mathbf{x}' \sim \mathcal{B}_p(\mathbf{x}, \tau)$. The expected value of this indicator gives the probability of disagreement under perturbation: $\mathbb{E}[k_\tau(\mathbf{x}')] = \Pr[m(\mathbf{x}') \neq a(\mathbf{x}')]$. Adversarial disagreement is challenging to control, especially in multi-class settings, as it involves evaluating probabilities over all possible output combinations of $m(\mathbf{x}')$ and $a(\mathbf{x}')$. To address this complexity, we next introduce a special case of adversarial disagreement—called *residual knowledge*—which offers a more tractable measure with operational meanings.

228 4.1 Connecting disagreement with residual knowledge

We focus on a particularly concerning form of adversarial disagreement, i.e., $1(m(\mathbf{x}') = y, a(\mathbf{x}') \neq y)$ for a forget sample $(\mathbf{x}, y) \in \mathcal{S}_f$. It implies that a forget sample, intended to be fully erased from the model, can be slightly perturbed such that the unlearned model still correctly classifies it, while the re-trained model does not. This discrepancy suggests that traces of the forget samples may remain embedded in the unlearned model’s decision boundary, even when formal indistinguishability guarantees are satisfied. This scenario motivates our definition of *residual knowledge*—the persistence of information about the forget set in the predictive behavior of the unlearned model. The presence of residual knowledge reveals a subtle yet significant vulnerability: it demonstrates how adversarial examples can compromise the effectiveness of unlearning, and underscores the need for stronger robustness guarantees that go beyond conventional (ϵ, δ) -unlearning.

Consider a forget sample $(\mathbf{x}, y) \in \mathcal{S}_f$, and let $m \sim M(A(\mathcal{S}), \mathcal{S}, \mathcal{S}_f)$ and $a \sim A(\mathcal{S}_r)$ be independently sampled unlearned and re-trained models, respectively. We formally define the residual knowledge around \mathbf{x} as the following non-negative ratio:

$$r_\tau((\mathbf{x}, y)) \triangleq \frac{\Pr[m(\mathbf{x}') = y]}{\Pr[a(\mathbf{x}') = y]}, \mathbf{x}' \stackrel{i.i.d.}{\sim} \mathcal{B}_p(\mathbf{x}, \tau). \quad (4)$$

This definition can be naturally extended to the entire forget set \mathcal{S}_f by averaging over all forget samples, $r_\tau(\mathcal{S}_f) \triangleq \frac{1}{|\mathcal{S}_f|} \sum_{(\mathbf{x}, y) \in \mathcal{S}_f} r_\tau((\mathbf{x}, y))$. Among the possible cases, $r_\tau(\mathcal{S}_f) > 1$ is especially concerning, as it indicates that the unlearned model m is more likely than the re-trained model a to correctly classify perturbed variants of forget samples. This suggests the presence of residual knowledge, which is the main privacy risk we aim to address in this paper. In this sense, the case where $r_\tau(\mathcal{S}_f) < 1$ is more tolerable, as it implies the unlearned model has lost the ability to recognize the forget samples. The case where $r_\tau(\mathcal{S}_f) = 1$ can only occur when $m = a$, meaning the unlearned model m achieves exact unlearning. In practice, $r_\tau((\mathbf{x}, y))$ can be estimated via Monte Carlo sampling. Specifically, by drawing c i.i.d. samples $\{\mathbf{x}'_i\}_{i=1}^c$ from $\mathcal{B}_p(\mathbf{x}, \tau)$, we approximate the ratio as $\hat{r}_\tau((\mathbf{x}, y)) = \frac{\sum_{i=1}^c \mathbb{1}(m(\mathbf{x}'_i) = y)}{\sum_{i=1}^c \mathbb{1}(a(\mathbf{x}'_i) = y)}$.

The notion of residual knowledge is closely tied to adversarial disagreement. In particular, residual knowledge offers both upper and lower bounds on the expected adversarial disagreement between the unlearned and re-trained models (cf. Lemma A.4):

$$r_\tau((\mathbf{x}, y)) \Pr[a(\mathbf{x}') = y] (1 - \Pr[a(\mathbf{x}') = y]) \leq \mathbb{E}[k_\tau(\mathbf{x}')] \leq 1 - r_\tau((\mathbf{x}, y)) \Pr[a(\mathbf{x}') = y]^2. \quad (5)$$

As $r_\tau((\mathbf{x}, y)) \rightarrow 1$, the expected adversarial disagreement $\mathbb{E}[k_\tau(\mathbf{x}')] approaches zero. In this sense, residual knowledge serves as a practical proxy for estimating the distinguishability between the unlearned and re-trained models, and provides a more tractable means to quantify and control adversarial disagreement.$

259 4.2 RURK: Robust unlearning against residual knowledge

To address residual knowledge, we propose a fine-tuning objective that simultaneously enforces unlearning and regulates residual knowledge. Ideally, residual knowledge should be close to 1; however, accurately computing $r_\tau(\mathcal{S}_f)$ requires access to a re-trained model $a \sim A(\mathcal{S}_r)$, which is typically unavailable during unlearning. Thus, our objective is to attenuate residual knowledge and ensure that $r_\tau(\mathcal{S}_f) \leq 1$. Recall that the numerator of the residual knowledge in Eq. (4) is $\Pr[m(\mathbf{x}') = y]$; thus, directly minimizing this probability alone can effectively suppress residual

⁴The randomness in $k(\mathbf{x}) = \mathbb{1}(M(\mathbf{x}) \neq A(\mathbf{x}))$ arises from the stochasticity of the models, whereas the randomness in $k_\tau(\mathbf{x}') = \mathbb{1}(m(\mathbf{x}') \neq a(\mathbf{x}'))$ comes from sampling adversarial examples $\mathbf{x}' \sim \mathcal{B}_p(\mathbf{x}, \tau)$.

knowledge, regardless of the denominator $\Pr[a(\mathbf{x}') = y]$. Based on this insight, we define the set of vulnerable perturbations as $\mathcal{V}((\mathbf{x}, y), \tau) \triangleq \{\mathbf{x}' \in \mathcal{B}_p(\mathbf{x}, \tau) | m(\mathbf{x}') = y\}$, which captures perturbed samples that continue to be associated with the true label by the unlearned model—indicating residual knowledge. In practice, we construct v such samples by adapting adversarial attack methods, such as FGSM or PGD, to solve the constrained minimization problem $\min_{\mathbf{z} \in \mathcal{B}_p(\mathbf{x}, \tau)} \ell(\mathbf{w}, (\mathbf{z}, y))$.

Given the vulnerable set, we formulate the following fine-tuning objective for RURK as:

$$L_{\text{RURK}}(\mathbf{w}, \mathcal{S}) = \underbrace{\frac{1}{|\mathcal{S}_r|} \sum_{(\mathbf{x}, y) \in \mathcal{S}_r} \ell(\mathbf{w}, (\mathbf{x}, y))}_{\text{Term (i)}} - \lambda \underbrace{\frac{1}{|\mathcal{S}_f|} \sum_{(\mathbf{x}, y) \in \mathcal{S}_f} \kappa(\mathbf{w}, (\mathbf{x}, y))}_{\text{Term (ii)}}, \quad (6)$$

where $\kappa(\mathbf{w}, (\mathbf{x}, y)) = \frac{1}{v} \sum_{\{\mathbf{x}'\}_{i=1}^v \in \mathcal{V}((\mathbf{x}, y), \tau)} \ell(\mathbf{w}, (\mathbf{x}', y))$, and $\lambda \geq 0$ is the regularization strength.

Term (i) preserves performance on the retain set, following prior work such as Neel et al. (2021); Kurmanji et al. (2024); Chien et al. (2024). Term (ii) serves as a regularization term that penalizes residual knowledge by discouraging the unlearned model from re-identifying vulnerable perturbations, thereby improving robustness with respect to residual knowledge. Explicitly searching for vulnerable perturbations in $\mathcal{V}((\mathbf{x}, y), \tau)$ during each gradient step can be computationally expensive. To mitigate this overhead, we may adopt a more efficient approximation by setting $\mathcal{V}((\mathbf{x}, y), \tau) = \mathcal{B}_p(\mathbf{x}, \tau)$, i.e., removing all label information in the neighborhood of each forget sample. We outline the RURK algorithm in Appendix B.

Note that as $\tau \rightarrow 0$, the objective in Eq. (6) reduces to that used in prior works such as Neel et al. (2021); Chien et al. (2024). When optimized via (Projected) Noisy Gradient Descent (NGD) (Chourasia et al., 2021), the resulting unlearned model satisfies (α, ϵ) -Rényi unlearning, where ϵ depends on the smoothness and Lipschitz continuity of the loss function $L(\mathbf{w}, \mathcal{S})$. Specifically, less smooth or less Lipschitz losses lead to larger ϵ , making the unlearned model more distinguishable from the re-trained one (Chien et al., 2024, Theorem 3.2). This reveals a fundamental trade-off between model indistinguishability and robustness against residual knowledge: increasing τ in term (ii) of Eq. (6) enlarges the adversarial perturbation space, which makes the adversarial loss $\kappa(\mathbf{w}, (\mathbf{x}, y))$ less smooth and with a larger Lipschitz constant—ultimately increasing ϵ .

5 Empirical Studies

We now evaluate residual knowledge of state-of-the-art unlearning algorithms and its mitigation via RURK on two vision benchmarks. We denote *Original* as the model $A(\mathcal{S})$ trained on the full dataset \mathcal{S} , and *Re-train* as the ideal (but not viable in real world) model $A(\mathcal{S}_r)$ re-trained from scratch without the forget set \mathcal{S}_f , used here as a reference. For experimental details, including dataset settings and hyper-parameter choices, refer to Appendix B; additional results are provided in Appendix C.

Data and unlearning settings. We evaluate unlearning methods on image classification using CIFAR-10 (Krizhevsky et al., 2009) with ResNet-18 (He et al., 2016), focusing on the random *sample unlearning* setting across two scenarios. The first scenario, small CIFAR-5, follows Golatkar et al. (2020a,b) and uses a reduced version of CIFAR-10 with 200 training and 200 test images per class from the first five classes. We randomly select 100 samples (50%) from class 0 as the forget set \mathcal{S}_f . The second scenario uses the full CIFAR-10 dataset, where 2,000 images (50%⁵) from class 0 are designated as \mathcal{S}_f . In both cases, only a subset of class 0 is unlearned, in contrast to the *class unlearning* setting (Kodge et al., 2024), which aims to remove all samples from a class. To avoid any external knowledge, both *Original* and *Re-train* models are trained from scratch without using any pre-trained weights. For additional experiments on class unlearning and with other model architectures such as VGG (Simonyan and Zisserman, 2014), see Appendix C.

Baseline algorithms. We evaluate three machine unlearning algorithms suited for small CIFAR-5 settings {CR, Fisher, NTK}, and eight state-of-the-art methods for the full CIFAR-10 {GD, NGD, GA, NegGrad+, EU-k, CF-k, SCRUB, SSD}. Certified Removal (CR) uses influence functions and a one-step Newton update to estimate and remove the effect of forget samples (Guo et al., 2020). This approach was extended by Golatkar et al. (2020a), who incorporate the Fisher information matrix (Fisher) computed on the retain set, and by Golatkar et al. (2020b), who apply Neural Tangent Kernel (NTK) linearization of neural networks. These pioneering methods, while foundational, are

⁵Each class contains 5,000 training images, with 20% held out for validation.

Table 1: Performance summary of various unlearning methods on image classification, including the proposed RURK, Original, Re-train, and 11 baseline approaches, evaluated under two unlearning scenarios: small CIFAR-5 and full CIFAR-10, both using ResNet-18. Results are reported in the format $a \pm b$, indicating the mean a and standard deviation b over 3 independent trials. The absolute performance gap relative to Re-train is shown in (blue). For methods that fail to recover the forget-set knowledge within 30 training epochs, the re-learn time is reported as “>30”.

Datasets	Methods	Evaluation Metrics					
		Retain Acc. (%)	Unlearn Acc. (%)	Test Acc. (%)	MIA Acc. (%)	Avg. Gap	Re-learn Time (# Epoch)
Small CIFAR-5	Original	99.93 \pm 0.10 (0.03)	0.00 \pm 0.00 (8.33)	95.37 \pm 0.80 (0.57)	4.67 \pm 3.30 (22.33)	7.82	-
	Re-train	99.96 \pm 0.05 (0.00)	8.33 \pm 3.30 (0.00)	94.80 \pm 0.85 (0.00)	27.00 \pm 5.66 (0.00)	0.00	3.33 \pm 0.47
	CR	99.56 \pm 0.47 (0.40)	14.00 \pm 5.66 (5.67)	91.80 \pm 0.99 (3.00)	58.17 \pm 0.79 (31.17)	10.06	-
	Fisher	92.67 \pm 0.63 (7.29)	12.67 \pm 0.94 (4.34)	88.80 \pm 1.98 (6.00)	47.33 \pm 6.13 (20.33)	9.49	3.00 \pm 1.41
	NTK	99.93 \pm 0.10 (0.03)	7.00 \pm 0.00 (1.33)	95.37 \pm 0.80 (0.57)	16.00 \pm 4.24 (11.00)	3.23	4.67 \pm 0.47
	RURK	99.52 \pm 0.37 (0.44)	5.67 \pm 2.36 (2.66)	93.83 \pm 0.90 (0.97)	33.33 \pm 12.26 (6.33)	2.60	2.00 \pm 0.00
CIFAR-10	Original	100.00 \pm 0.00 (0.00)	0.00 \pm 0.00 (9.47)	94.76 \pm 0.05 (1.46)	0.07 \pm 0.02 (22.43)	8.34	-
	Re-train	100.00 \pm 0.00 (0.00)	9.47 \pm 0.61 (0.00)	93.30 \pm 0.20 (0.00)	22.50 \pm 0.60 (0.00)	0.00	17.33 \pm 6.65
	GD	99.98 \pm 0.01 (0.02)	0.00 \pm 0.00 (9.47)	94.29 \pm 0.07 (0.99)	0.10 \pm 0.00 (22.4)	8.22	0.20 \pm 0.04
	NGD	97.53 \pm 0.03 (2.47)	10.67 \pm 0.61 (1.20)	90.70 \pm 0.11 (2.60)	3.70 \pm 0.53 (18.80)	6.27	20.67 \pm 14.61
	GA	95.41 \pm 0.04 (4.59)	61.37 \pm 0.17 (51.91)	85.98 \pm 0.11 (7.32)	0.00 \pm 0.00 (22.5)	21.25	1.00 \pm 0.00
	NegGrad+	99.28 \pm 0.01 (0.72)	14.00 \pm 0.44 (4.53)	92.02 \pm 0.02 (1.28)	18.18 \pm 0.43 (4.32)	2.71	1.00 \pm 0.00
	EU-k	99.34 \pm 0.03 (0.66)	1.12 \pm 0.08 (8.35)	92.97 \pm 0.08 (0.33)	0.87 \pm 0.10 (21.63)	7.74	4.33 \pm 3.40
	CF-k	100.00 \pm 0.00 (0.00)	0.00 \pm 0.00 (9.47)	94.38 \pm 0.04 (1.08)	0.42 \pm 0.10 (22.08)	8.16	1.00 \pm 0.00
	SCRUB	99.61 \pm 0.02 (0.39)	12.45 \pm 0.29 (2.98)	92.70 \pm 0.03 (0.60)	7.10 \pm 0.14 (15.40)	4.84	> 30
	SSD	96.49 \pm 0.02 (3.51)	66.45 \pm 0.82 (56.98)	88.59 \pm 0.06 (4.71)	5.12 \pm 0.44 (17.38)	20.65	7.33 \pm 0.47
	RURK	99.55 \pm 0.07 (0.45)	14.63 \pm 2.17 (5.16)	92.60 \pm 0.24 (0.70)	18.20 \pm 2.47 (4.30)	2.65	> 30

not scalable to large models due to the computational cost of Hessian-based operations. Gradient Descent (GD) fine-tunes Original on the retain set \mathcal{S}_r using standard SGD (Neel et al., 2021). Noisy Gradient Descent (NGD) simply modifies GD by adding Gaussian noise in the gradient update steps for better privacy guarantees (Chourasia and Shah, 2023; Chien et al., 2024). On the other hand, Gradient Ascent (GA) removes the influence of \mathcal{S}_f by reversing gradient updates on the forget set (Graves et al., 2021; Jang et al., 2022). NegGrad+ combines both strategies by applying GD to \mathcal{S}_r and GA to \mathcal{S}_f simultaneously (Kurmanji et al., 2024). To improve parameter efficiency, Goel et al. (2022) propose two layer-wise methods: Exact Unlearning the last k layers (EU- k), which re-trains them from scratch, and Catastrophically Forgetting the last k layers (CF- k), which only fine-tunes them on \mathcal{S}_r . SCRUB casts unlearning as a teacher-student distillation process, where the student selectively learns retain-set knowledge from the teacher (Kurmanji et al., 2024). The final method, SSD, selectively dampens model weights by uses Fisher information to estimate the influence of \mathcal{S}_f (Foster et al., 2024), a scalable version of Fisher. Additional details about these baselines are provided in Appendix B.

Evaluation metrics. We adopt five evaluation metrics, as described in § 2.2, to comprehensively assess both the unlearning efficacy and the utility of the resulting models. To capture different dimensions of unlearning effectiveness, we first report Unlearning Accuracy⁶, following Fan et al. (2023). Second, we conduct a MIA using a support vector classifier (SVC) trained to distinguish training samples from test samples based on the model’s output likelihoods; we report the SVC’s attack failure rate (MIA Accuracy) as a measure of privacy protection. Third, Re-learn Time quantifies how easily a model can re-acquire the forget-set information: it is defined as the number of fine-tuning epochs needed for the model m to satisfy $L(m, \mathcal{S}_f) \leq (1 + \eta)L(\text{Original}, \mathcal{S}_f)$, with $\eta = 0.05$. To assess model utility, we report classification accuracy on the retain set and the hold-out test set, referred to as Retain Accuracy and Test Accuracy, capturing both performance preservation and generalization. As discussed in § 2.1, a desirable unlearning method should minimize the deviation of performance from Re-train. To this end, we compute the Average Gap (Avg. Gap)—the mean absolute gap from Re-train across Retain, Unlearn, Test, and MIA accuracies—where a smaller Avg. Gap indicates better unlearning. Notably, methods such as CR, Fisher, NTK, and NGD already offer certified unlearning guarantees under frameworks like (ϵ, δ) -unlearning, Rényi unlearning, or bounded KL divergence; these guarantees do not necessarily imply a small performance gap on accuracy-based metrics.

Performance comparison. Table 1 compares the performance of Original, the ideal reference Re-train, our proposed RURK, and 11 baseline unlearning methods across two unlearning scenarios. RURK achieves the smallest average performance gap to Re-train in both settings. We apply SGD

⁶Unlearning accuracy is defined as $1 - \text{forget accuracy}$, i.e., the classification error on the forget set \mathcal{S}_f .

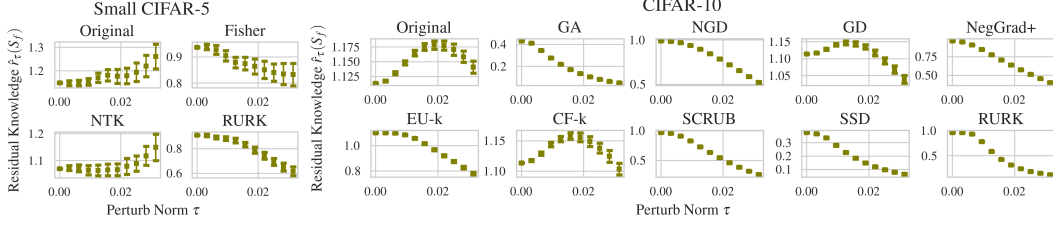


Figure 2: Residual knowledge $\hat{r}_\tau(S_f)$ of the proposed RURK, Original, and other unlearning methods across two unlearning scenarios, evaluated under varying perturbation norms τ .

with $\tau = 0.03$ and $v = 1$ in Term (ii) of Eq. (6), making RURK’s time complexity comparable to other gradient-based approaches such as GD, NGD, and NegGrad+. In the small CIFAR-5 scenario, NTK shows a competitive Avg. Gap but suffers from the highest MIA accuracy, indicating vulnerability to privacy attacks. Fisher over-forgets the target samples, significantly degrading utility, as seen in its low retain and test accuracies. For CIFAR-10, NegGrad+ performs similarly to RURK in Avg. Gap but shows a much shorter re-learn time than Re-train, implying an implicit retention of forget-set knowledge. GA and SSD aggressively erase forget-set information, achieving high unlearn accuracy but at the cost of test accuracy dropping below 90%. Both EU-k and CF-k (with $k = 5$) re-train or fine-tune only the linear and last two residual blocks, but the remaining layers of ResNet-18 still retain substantial information about the forget set. SCRUB has the closest overall performance to RURK when including re-learn time, but unlike RURK, it requires extra memory to store a student model, whereas RURK supports in-place updates. We defer the performance of other unlearning baselines on small CIFAR-5, and an ablation study on the hyper-parameters (e.g., τ , v and λ) in RURK to Appendix C.

Residual knowledge. Figure 2 presents⁷ estimates of residual knowledge $\hat{r}_\tau((\mathbf{x}, y))$ under varying perturbation radius τ , using Gaussian noise ($p = 2$) to draw $c = 100$ samples from $\mathcal{B}_p(\mathbf{x}, \tau)$; cf. Eq. (4). As expected, Original consistently exhibits residual knowledge greater than 1. In small CIFAR-5, although NTK performs comparably to RURK in Avg. Gap and re-learn time (Table 1), its residual knowledge remains above 1, exposing privacy risks. This highlights residual knowledge as a complementary and necessary metric alongside unlearn accuracy, MIA, and re-learn time. In contrast, RURK keeps residual knowledge close to 1 for $\tau < 0.01$ and successfully suppresses it for larger τ , as intended. For CIFAR-10, GD and CF-k inherit the high residual knowledge of Original, showing ineffectiveness in removing forget-set information. Conversely, GA and SSD yield residual knowledge well below 1 even at $\tau = 0.00$ (i.e., forget samples themselves), again pointing to excessive unlearning. Methods like NGD, NegGrad+, EU-k, and SCRUB follow trends similar to RURK, but RURK achieves more stable control of the residual knowledge near $\tau = 0.01$ and stronger suppression when $\tau > 0.01$. For the estimates of residual knowledge with other attacks such as FGSM, see Appendix C.

6 Final remark

Limitations. Ideally, unlearned and re-trained models should be statistically indistinguishable not only on original inputs but also on all perturbations within $\mathcal{B}_p(\mathbf{x}, \tau)$. However, controlling adversarial disagreement across such perturbations is computationally infeasible—it is as hard as achieving perfect adversarial robustness. Moreover, since the re-trained model is typically unavailable during unlearning, we can only bound one side of the residual knowledge (i.e., $r_\tau(S_f) \leq 1$; see § 4.2).

Future directions. First, our probabilistic analysis in § 3 could be extended to account for hypothesis class complexity and linked to the transferability of adversarial examples (Tramèr et al., 2017). Second, the indistinguishability–robustness trade-off introduced in § 4.2 opens up new directions in both unlearning and adversarial robustness, warranting deeper investigation. Third, our mitigation objective (cf. Eq. (6)) resembles a reverse of adversarial training. This raises an intriguing question: does adversarial training or distributional robust optimization make a model harder to unlearn or increase its residual knowledge?

Broader impact. Residual knowledge introduces new privacy risks. For instance, if a user opts out of a biometric-based payment system (e.g., palm or facial recognition), residual traces may still allow adversaries to craft perturbed inputs that the system accepts—potentially enabling unauthorized access. Such vulnerabilities undermine trust in ML systems and challenge current interpretations of the “Right to be Forgotten.”

⁷By Eq. (4), the residual knowledge of Re-train is exactly 1 across all τ ; therefore we skip it in the figure.

References

- Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. (2016). Deep learning with differential privacy. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*.
- Belrose, N., Schneider-Joseph, D., Ravfogel, S., Cotterell, R., Raff, E., and Biderman, S. (2023). Leace: Perfect linear concept erasure in closed form. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.
- Bourtole, L., Chandrasekaran, V., Choquette-Choo, C. A., Jia, H., Travers, A., Zhang, B., Lie, D., and Papernot, N. (2021). Machine unlearning. In *Proceedings of the IEEE Symposium on Security and Privacy (SSP)*.
- Brophy, J. and Lowd, D. (2021). Machine unlearning for random forests. In *Proceedings of the International Conference on Machine Learning (ICML)*.
- Bun, M., Gaboardi, M., Hopkins, M., Impagliazzo, R., Lei, R., Pitassi, T., Sivakumar, S., and Sorrell, J. (2023). Stability is stable: Connections between replicability, privacy, and adaptive generalization. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing (STOC)*.
- Cao, Y. and Yang, J. (2015). Towards making systems forget with machine unlearning. In *Proceedings of the IEEE Symposium on Security and Privacy (SSP)*.
- Carlini, N., Hayes, J., Nasr, M., Jagielski, M., Schweg, V., Tramer, F., Balle, B., Ippolito, D., and Wallace, E. (2023). Extracting training data from diffusion models. In *Proceedings of the 32nd USENIX Security Symposium*.
- Chien, E., Wang, H., Chen, Z., and Li, P. (2024). Langevin unlearning: A new perspective of noisy gradient descent for machine unlearning. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.
- Chourasia, R. and Shah, N. (2023). Forget unlearning: Towards true data-deletion in machine learning. In *Proceedings of the International Conference on Machine Learning (ICML)*.
- Chourasia, R., Ye, J., and Shokri, R. (2021). Differential privacy dynamics of langevin diffusion and noisy gradient descent. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.
- Dwork, C. and Roth, A. (2014). The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407.
- Eldan, R. and Russinovich, M. (2023). Who’s harry potter? approximate unlearning in LLMs. *arXiv preprint arXiv:2310.02238*.
- Fan, C., Liu, J., Zhang, Y., Wong, E., Wei, D., and Liu, S. (2023). Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation. In *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Foster, J., Schoepf, S., and Brintrup, A. (2024). Fast machine unlearning without retraining through selective synaptic dampening. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
- French, R. M. (1999). Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135.
- Gandikota, R., Materzynska, J., Fiotto-Kaufman, J., and Bau, D. (2023). Erasing concepts from diffusion models. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2426–2436.
- Ginart, A., Guan, M., Valiant, G., and Zou, J. Y. (2019). Making AI forget you: Data deletion in machine learning. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

439 Goel, S., Prabhu, A., Sanyal, A., Lim, S.-N., Torr, P., and Kumaraguru, P. (2022). Towards adversarial
440 evaluations for inexact machine unlearning. *arXiv preprint arXiv:2201.06640*.

441 Golatkar, A., Achille, A., and Soatto, S. (2020a). Eternal sunshine of the spotless net: Selective
442 forgetting in deep networks. In *Proceedings of the IEEE Conference on Computer Vision and*
443 *Pattern Recognition (CVPR)*.

444 Golatkar, A., Achille, A., and Soatto, S. (2020b). Forgetting outside the box: Scrubbing deep
445 networks of information accessible from input-output observations. In *Proceedings of the European*
446 *Conference on Computer Vision (ECCV)*.

447 Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep learning*. MIT press
448 Cambridge.

449 Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples.
450 *arXiv preprint arXiv:1412.6572*.

451 Graves, L., Nagisetty, V., and Ganesh, V. (2021). Amnesiac machine learning. In *Proceedings of the*
452 *AAAI Conference on Artificial Intelligence*.

453 Gross, L. (1975). Logarithmic sobolev inequalities. *American Journal of Mathematics*, 97(4):1061–
454 1083.

455 Guo, C., Goldstein, T., Hannun, A., and Van Der Maaten, L. (2020). Certified data removal from
456 machine learning models. In *Proceedings of the International Conference on Machine Learning*
457 *(ICML)*.

458 He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In
459 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

460 Hinton, G., Vinyals, O., and Dean, J. (2014). Distilling the knowledge in a neural network. In
461 *Proceedings of Advances in Neural Information Processing Systems (NeurIPS), Deep Learning*
462 *Workshop*.

463 Impagliazzo, R., Lei, R., Pitassi, T., and Sorrell, J. (2022). Reproducibility in learning. In *Proceedings*
464 *of the 54th annual ACM symposium on theory of computing (STOC)*.

465 Izzo, Z., Smart, M. A., Chaudhuri, K., and Zou, J. (2021). Approximate data deletion from machine
466 learning models. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

467 Jang, J., Yoon, D., Yang, S., Cha, S., Lee, M., Logeswaran, L., and Seo, M. (2022). Knowledge
468 unlearning for mitigating privacy risks in language models. *arXiv preprint arXiv:2210.01504*.

469 Kalavasis, A., Karbasi, A., Moran, S., and Velegkas, G. (2023). Statistical indistinguishability of
470 learning algorithms. In *Proceedings of the International Conference on Machine Learning (ICML)*.

471 Kim, H. (2020). Torchattacks: A pytorch repository for adversarial attacks. *arXiv preprint*
472 *arXiv:2010.01950*.

473 Kodge, S., Saha, G., and Roy, K. (2024). Deep unlearning: Fast and efficient gradient-free class
474 forgetting. *Transactions on Machine Learning Research (TMLR)*.

475 Krishna, S., Han, T., Gu, A., Wu, S., Jabbari, S., and Lakkaraju, H. (2025). The disagreement problem
476 in explainable machine learning: A practitioner’s perspective. *Transactions on Machine Learning*
477 *Research (TMLR)*.

478 Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.
479 *Technical Report*.

480 Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *The annals of mathematical*
481 *statistics*, 22(1):79–86.

482 Kulynych, B., Hsu, H., Troncoso, C., and Calmon, F. P. (2023). Arbitrary decisions are a hidden cost
483 of differentially private training. In *Proceedings of the ACM Conference on Fairness, Accountability,*
484 *and Transparency (FAccT)*.

485 Kurmanji, M., Triantafillou, P., Hayes, J., and Triantafillou, E. (2024). Towards unbounded machine
486 unlearning. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

487 Li, G., Hsu, H., Marculescu, R., et al. (2024). Machine unlearning for image-to-image generative
488 models. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

489 Liu, S., Yao, Y., Jia, J., Casper, S., Baracaldo, N., Hase, P., Yao, Y., Liu, C. Y., Xu, X., Li, H., et al.
490 (2025). Rethinking machine unlearning for large language models. *Nature Machine Intelligence*,
491 pages 1–14.

492 Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2018). Towards deep learning
493 models resistant to adversarial attacks. In *Proceedings of the International Conference on Learning
494 Representations (ICLR)*.

495 Marcel, S. and Rodriguez, Y. (2010). Torchvision the machine-vision package of torch. In *Proceedings
496 of the 18th ACM International Conference on Multimedia (MM)*.

497 Marchant, N. G., Rubinstein, B. I., and Alfeld, S. (2022). Hard to forget: Poisoning attacks on
498 certified machine unlearning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

499 Martens, J. (2020). New insights and perspectives on the natural gradient method. *Journal of Machine
500 Learning Research (JMLR)*, 21(146):1–76.

501 Milman, V. D. and Schechtman, G. (1986). *Asymptotic theory of finite dimensional normed spaces:
502 Isoperimetric inequalities in riemannian manifolds*, volume 1200. Springer Science & Business
503 Media.

504 Mironov, I. (2017). Rényi differential privacy. In *Proceedings of the IEEE Computer Security
505 Foundations Symposium (CSF)*.

506 Neel, S., Roth, A., and Sharifi-Malvajerdi, S. (2021). Descent-to-delete: Gradient-based methods for
507 machine unlearning. In *Proceedings of Algorithmic Learning Theory (ALT)*.

508 Nguyen, Q. P., Low, B. K. H., and Jaillet, P. (2020). Variational bayesian unlearning. In *Proceedings
509 of Advances in Neural Information Processing Systems (NeurIPS)*.

510 Nguyen, T. T., Huynh, T. T., Ren, Z., Nguyen, P. L., Liew, A. W.-C., Yin, H., and Nguyen, Q. V. H.
511 (2022). A survey of machine unlearning. *arXiv preprint arXiv:2209.02299*.

512 Paszke, A. (2019). Pytorch: An imperative style, high-performance deep learning library. *arXiv
513 preprint arXiv:1912.01703*.

514 Pawelczyk, M., Di, J. Z., Lu, Y., Kamath, G., Sekhari, A., and Neel, S. (2025). Machine unlearning
515 fails to remove data poisoning attacks. In *Proceedings of the International Conference on Learning
516 Representations (ICLR)*.

517 Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M.,
518 Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python.
519 *the Journal of machine Learning research (JMLR)*, 12:2825–2830.

520 Ravfogel, S., Twiton, M., Goldberg, Y., and Cotterell, R. D. (2022). Linear adversarial concept
521 erasure. In *Proceedings of the International Conference on Machine Learning (ICML)*.

522 Rényi, A. (1961). On measures of entropy and information. In *Proceedings of the 4th Berkeley
523 Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of
524 Statistics*.

525 Schelter, S., Grafberger, S., and Dunning, T. (2021). Hedgecut: Maintaining randomised trees for
526 low-latency machine unlearning. In *Proceedings of the International Conference on Management
527 of Data*.

528 Sekhari, A., Acharya, J., Kamath, G., and Suresh, A. T. (2021). Remember what you want to forget:
529 Algorithms for machine unlearning. In *Proceedings of Advances in Neural Information Processing
530 Systems (NeurIPS)*.

531 Shastri, S., Wasserman, M., and Chidambaram, V. (2019). The seven sins of personal-data processing
532 systems under GDPR. In *Proceedings of the 11th USENIX Workshop on Hot Topics in Cloud*
533 *Computing*.

534 Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image
535 recognition. *arXiv preprint arXiv:1409.1556*.

536 Talagrand, M. (1995). Concentration of measure and isoperimetric inequalities in product spaces.
537 *Publications Mathématiques de l’Institut des Hautes Etudes Scientifiques*, 81:73–205.

538 Tramèr, F., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. (2017). The space of
539 transferable adversarial examples. *arXiv preprint arXiv:1704.03453*.

540 Uma, A. N., Fornaciari, T., Hovy, D., Paun, S., Plank, B., and Poesio, M. (2021). Learning from
541 disagreement: A survey. *Journal of Artificial Intelligence Research (JAIR)*, 72:1385–1470.

542 Voigt, P. and Von dem Bussche, A. (2017). The EU general data protection regulation (GDPR). *A*
543 *Practical Guide, 1st Ed., Cham: Springer International Publishing*, 10(3152676):10–5555.

544 Wang, L., Chen, T., Yuan, W., Zeng, X., Wong, K.-F., and Yin, H. (2023). KGA: A general machine
545 unlearning framework based on knowledge gap alignment. In *Proceedings of the 61st Annual*
546 *Meeting of the Association for Computational Linguistics (ACL)*.

547 Wang, W., Tian, Z., Zhang, C., and Yu, S. (2024). Machine unlearning: A comprehensive survey.
548 *arXiv preprint arXiv:2405.07406*.

549 Wu, Y., Dobriban, E., and Davidson, S. (2020). Deltagrad: Rapid retraining of machine learning
550 models. In *Proceedings of the International Conference on Machine Learning (ICML)*.

551 Xu, H., Zhu, T., Zhang, L., Zhou, W., and Yu, P. S. (2023). Machine unlearning: A survey. *ACM*
552 *Computing Surveys*, 56(1).

553 Yao, J., Chien, E., Du, M., Niu, X., Wang, T., Cheng, Z., and Yue, X. (2024). Machine unlearning of
554 pre-trained large language models. In *Proceedings of the 62nd Annual Meeting of the Association*
555 *for Computational Linguistics (ACL)*.

556 Zhang, B., Dong, Y., Wang, T., and Li, J. (2024). Towards certified unlearning for deep neural
557 networks. In *Proceedings of the International Conference on Machine Learning (ICML)*.

558 Zhang, R. and Zhang, S. (2022). Rethinking influence functions of neural networks in the over-
559 parameterized regime. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

560 Zhao, C., Qian, W., Li, Y., Li, W., and Huai, M. (2024). Rethinking adversarial robustness in the
561 context of the right to be forgotten. In *Proceedings of the International Conference on Machine*
562 *Learning (ICML)*.

563 The appendix is divided into the following parts. Appendix A: Omitted proofs and theoretical results;
 564 Appendix B: Details on the experimental setup; and Appendix C: Additional empirical results and
 565 ablation studies.

566 A Omitted proofs and theoretical results

567 We first introduce (and prove) the following useful lemmas to facilitate the proofs of the propositions.
 568 The first lemma is a variant of the (ϵ, δ) -indistinguishability in Definition 1.

569 **Lemma A.1** (Probabilistic indistinguishability). *If X and Y are (ϵ, δ) -indistinguishable, then with*
 570 *probability at least $1 - 2\delta/(1 - e^{-\epsilon})$ over z drawn from $\text{support}(X) \cup \text{support}(Y)$, we have*

$$e^{-2\epsilon} \Pr[Y = z] \leq \Pr[X = z] \leq e^{2\epsilon} \Pr[Y = z]. \quad (\text{A.1})$$

571 *Proof.* We first consider the complement event of the right inequality in Eq. (A.1). Define $\mathcal{Z} =$
 572 $\{z \mid \Pr[X = z] > e^{2\epsilon} \Pr[Y = z]\}$, we directly have $\Pr[X \in \mathcal{Z}] > e^{2\epsilon} \Pr[Y \in \mathcal{Z}]$. By (ϵ, δ) -
 573 indistinguishable, we have

$$\begin{aligned} \delta &\geq \Pr[X \in \mathcal{Z}] - e^\epsilon \Pr[Y \in \mathcal{Z}] > (e^{2\epsilon} - e^\epsilon) \Pr[Y \in \mathcal{Z}] \\ \Rightarrow \Pr[Y \in \mathcal{Z}] &< \frac{\delta}{e^{2\epsilon} - e^\epsilon} = \frac{\delta}{e^{2\epsilon}(1 - e^{-\epsilon})}. \end{aligned} \quad (\text{A.2})$$

574 Now consider the complement event of the left inequality in Eq. (A.1) and define $\mathcal{Z}' = \{z \mid \Pr[X =$
 575 $z] < e^{-2\epsilon} \Pr[Y = z]\}$, by similar algebra, we have

$$\begin{aligned} \delta &\geq \Pr[Y \in \mathcal{Z}'] - e^\epsilon \Pr[X \in \mathcal{Z}'] > (e^{2\epsilon} - e^\epsilon) \Pr[X \in \mathcal{Z}'] \\ \Rightarrow \Pr[X \in \mathcal{Z}'] &< \frac{\delta}{e^{2\epsilon} - e^\epsilon} \\ \Rightarrow \Pr[Y \in \mathcal{Z}'] &\leq e^\epsilon \Pr[X \in \mathcal{Z}'] + \delta < e^\epsilon \frac{\delta}{e^{2\epsilon} - e^\epsilon} + \delta = \frac{\delta}{1 - e^{-\epsilon}}. \end{aligned} \quad (\text{A.3})$$

576 Since $\frac{\delta}{1 - e^{-\epsilon}}$ is always larger than $\frac{\delta}{e^{2\epsilon}(1 - e^{-\epsilon})}$ (by a factor of $e^{2\epsilon}$), by combining Eq. (A.2) and
 577 Eq. (A.3), we have

$$\begin{aligned} \Pr[Y \in \mathcal{Z} \cup \mathcal{Z}'] &= \Pr[\{\Pr[X = z] > e^{2\epsilon} \Pr[Y = z]\} \text{ or } \{\Pr[X = z] < e^{-2\epsilon} \Pr[Y = z]\}] \\ &\leq \frac{\delta}{1 - e^{-\epsilon}}. \end{aligned} \quad (\text{A.4})$$

578 With the same analysis on $\Pr[X \in \mathcal{Z}]$ and $\Pr[X \in \mathcal{Z}']$, we have $\Pr[X \in \mathcal{Z} \cup \mathcal{Z}'] \leq \frac{\delta}{1 - e^{-\epsilon}}$. Finally,
 579 putting together the the probability bounds on $\Pr[Y \in \mathcal{Z} \cup \mathcal{Z}']$ and $\Pr[X \in \mathcal{Z} \cup \mathcal{Z}']$, we have

$$\Pr[e^{-2\epsilon} \Pr[Y = z] \leq \Pr[X = z] \leq e^{2\epsilon} \Pr[Y = z]] \leq 1 - \frac{\delta}{1 - e^{-\epsilon}}, \quad (\text{A.5})$$

580 as desired. \square

581 The second lemma relates to the isoperimetric inequality (Talagrand, 1995). Before stating it, we
 582 define the notion of a τ -expansion of a set. Given a set $\mathcal{C} \subset \mathbb{R}^d$, its τ -expansion with respect to the
 583 ℓ_p -norm is defined as

$$\mathcal{E}(\mathcal{C}, p, \tau) \triangleq \{\mathbf{c} \in \mathcal{R}^d \mid \|\mathbf{c} - \mathbf{g}\|_p \leq \tau \text{ for some } \mathbf{g} \in \mathcal{C}\}. \quad (\text{A.6})$$

584 The isoperimetric inequality is used to characterize the normalized surface area of the τ -expansion
 585 of a half unit sphere, showing how it grows with the expansion radius τ and the dimension d . This
 586 property is leveraged in the condition of Proposition 2, and is formally stated below.

587 **Lemma A.2** (Milman and Schechtman (1986)). *Let \mathcal{C} be the half unit sphere in \mathbb{R}^d , i.e., $\mathcal{C} \in \mathbb{S}^{d-1}$*
 588 *and $\text{surf}(\mathcal{C})/\text{surf}(\mathbb{S}^{d-1}) \geq 1/2$, then with $p = 2$ or $p = \infty$, the τ -expansion \mathcal{C} has the surface area*
 589 *that satisfies*

$$\frac{\text{surf}(\mathcal{E}(\mathcal{C}, p, \tau))}{\text{surf}(\mathbb{S}^{d-1})} \geq 1 - \left(\frac{\pi}{8}\right)^{1/2} \exp\left(-\frac{d-1}{2}\tau^2\right). \quad (\text{A.7})$$

590 The third lemma utilizes Lemma A.1 to lower bound the probability of disagreement, i.e., the expected
 591 value of the disagreement indicator function $k(\mathbf{x})$.

592 **Lemma A.3** (Lower bound on disagreement probability). *If the unlearned $M(A(\mathcal{S}), \mathcal{S}, \mathcal{S}_f)$ and re-*
 593 *trained $A(\mathcal{S}_r)$ models satisfies (ϵ, δ) -unlearning, then with probability $2\delta/(1 - e^{-\epsilon})$, the probability*
 594 *of disagreement among the two models on a sample $\mathbf{x} \in \mathcal{S}$ is lower bounded by*

$$\mathbb{E}[k(\mathbf{x})] = \mathbb{E}[\mathbb{1}(M(\mathbf{x}) \neq A(\mathbf{x}))] = \Pr[M(\mathbf{x}) \neq A(\mathbf{x})] > 1 - \mathcal{O}(e^{-2\epsilon}). \quad (\text{A.8})$$

595 *Proof.* We prove the lower bound by directly decompose $\Pr[M(\mathbf{x}) \neq A(\mathbf{x})]$, i.e.,

$$\begin{aligned} \Pr[M(\mathbf{x}) \neq A(\mathbf{x})] &= \int_{h \in \mathcal{H}} \Pr[m(\mathbf{x}) \neq a(\mathbf{x}) | M = m = h, A = a \neq h] \Pr[M \neq h, A = h] dh \\ &\quad + \int_{h \in \mathcal{H}} \Pr[m(\mathbf{x}) \neq a(\mathbf{x}) | M = m = h, A = a = h] \Pr[M = h, A = h] dh \end{aligned} \quad (\text{A.9})$$

596 The second integral in Eq. (A.9) is zero since $\Pr[m(\mathbf{x}) \neq a(\mathbf{x}) | M = m = h, A = a = h] = 0$.
 597 Moreover, by Lemma A.1, with probability $1 - 2\delta/(1 - e^{-\epsilon})$, we have

$$\Pr[M \neq h, A = h] = \Pr[M \neq h] \Pr[A = h] \leq \Pr[A = h] - e^{-2\epsilon} \Pr[A = h]^2. \quad (\text{A.10})$$

598 In other words, with $2\delta/(1 - e^{-\epsilon})$, we have

$$\Pr[M \neq h, A = h] = \Pr[M \neq h] \Pr[A = h] > \Pr[A = h] - e^{-2\epsilon} \Pr[A = h]^2. \quad (\text{A.11})$$

599 The combination of Eq. (A.9) and Eq. (A.11) yield

$$\begin{aligned} \Pr[M(\mathbf{x}) \neq A(\mathbf{x})] &= \int_{h \in \mathcal{H}} \Pr[m(\mathbf{x}) \neq a(\mathbf{x}) | M = m = h, A = a \neq h] \Pr[M \neq h, A = h] dh \\ &> \int_{h \in \mathcal{H}} \Pr[A = h] - e^{-2\epsilon} \Pr[A = h]^2 dh \\ &= \int_{h \in \mathcal{H}} \Pr[A = h] dh - e^{-2\epsilon} \int_{h \in \mathcal{H}} \Pr[A = h]^2 dh \\ &= 1 - \mathcal{O}(e^{-2\epsilon}), \end{aligned} \quad (\text{A.12})$$

600 the desired result, as $\int_{h \in \mathcal{H}} \Pr[A = h] dh = 1$ and $\int_{h \in \mathcal{H}} \Pr[A = h]^2 dh$ is a constant. \square

601 The fourth lemma provides the relation between the adversarial disagreement $k_\tau(\mathbf{x}') = \mathbb{1}(M(\mathbf{x}') \neq$
 602 $A(\mathbf{x}'))$ and the residual knowledge $r_\tau((\mathbf{x}, y))$.

603 **Lemma A.4** (Bounding adversarial disagreement with the residual knowledge). *The expected value*
 604 *of $k_\tau(\mathbf{x}')$ over the perturbation distribution is upper and lower bounded by*

$$r_\tau((\mathbf{x}, y)) \Pr[a(\mathbf{x}') = y] (1 - \Pr[a(\mathbf{x}') = y]) \leq \mathbb{E}[k_\tau(\mathbf{x}')] \leq 1 - r_\tau((\mathbf{x}, y)) \Pr[a(\mathbf{x}') = y]^2. \quad (\text{A.13})$$

605 *Proof.* We prove the lemma by directly following the definition of adversarial disagreement. First,
 606 we have

$$\begin{aligned} \mathbb{E}[k_\tau(\mathbf{x}')] &= \Pr[m(\mathbf{x}') \neq a(\mathbf{x}')] = \sum_{l \in \mathcal{Y}} \Pr[m(\mathbf{x}') = l, a(\mathbf{x}') \neq l] \\ &= \sum_{l \in \mathcal{Y}} \Pr[m(\mathbf{x}') = l] (1 - \Pr[a(\mathbf{x}') = l]) \geq \Pr[m(\mathbf{x}') = y] (1 - \Pr[a(\mathbf{x}') = y]), \end{aligned} \quad (\text{A.14})$$

607 where the final inequality follows by isolating the contribution of the true label y . By substituting the
 608 definition of residual knowledge from Eq. (4), we obtain

$$\mathbb{E}[k_\tau(\mathbf{x}')] \geq r_\tau((\mathbf{x}, y)) \Pr[a(\mathbf{x}') = y] (1 - \Pr[a(\mathbf{x}') = y]). \quad (\text{A.15})$$

609 Similarly, by evaluating $1 - \mathbb{E}[k_\tau(\mathbf{x}')] = \Pr[m(\mathbf{x}') = a(\mathbf{x}')]$, we have

$$\begin{aligned} 1 - \mathbb{E}[k_\tau(\mathbf{x}')] &= \Pr[m(\mathbf{x}') = a(\mathbf{x}')] = \sum_{l \in \mathcal{Y}} \Pr[m(\mathbf{x}') = l, a(\mathbf{x}') = l] \\ &\geq \Pr[m(\mathbf{x}') = y] \Pr[a(\mathbf{x}') = y] = r_\tau((\mathbf{x}, y)) \Pr[a(\mathbf{x}') = y]^2 \\ &\Rightarrow \mathbb{E}[k_\tau(\mathbf{x}')] \leq 1 - r_\tau((\mathbf{x}, y)) \Pr[a(\mathbf{x}') = y]^2. \end{aligned} \quad (\text{A.16})$$

610

□

611 A.1 Proof of Proposition 1

612 Suppose the unlearned $M(A(\mathcal{S}), \mathcal{S}, \mathcal{S}_f)$ and re-trained $A(\mathcal{S}_r)$ models are (ϵ, δ) -indistinguishable,
613 and let \mathbf{x} be a fixed sample. From Lemma A.1, we have that with probability $2\delta/(1 - e^\epsilon)$, h drawn
614 from $\text{support}(M) \cup \text{support}(A)$, we have

$$\Pr[M = h] \leq e^{-2\epsilon} \Pr[A = h] \text{ or } e^{2\epsilon} \Pr[A = h] \leq \Pr[M = h]. \quad (\text{A.17})$$

615 Therefore, for all $\mathcal{X}' \subseteq \mathcal{X}$, by using the right inequality in Eq. (A.17),

$$\begin{aligned} \Pr[g_{\mathbf{x}}(m) \in \mathcal{X}'] &= \Pr[g_{\mathbf{x}}(h) \in \mathcal{X}', M = h] = \Pr[g_{\mathbf{x}}(h) \in \mathcal{X}'] \Pr[M = h] \\ &\geq e^{2\epsilon} \Pr[g_{\mathbf{x}}(h) \in \mathcal{X}'] \Pr[A = h] = e^{2\epsilon} \Pr[g_{\mathbf{x}}(a) \in \mathcal{X}']. \end{aligned} \quad (\text{A.18})$$

616 Similarly, for the other inequality, we have

$$\Pr[g_{\mathbf{x}}(m) \in \mathcal{X}'] \leq e^{-2\epsilon} \Pr[g_{\mathbf{x}}(h) \in \mathcal{X}'] \Pr[A = h] = e^{-2\epsilon} \Pr[g_{\mathbf{x}}(a) \in \mathcal{X}']. \quad (\text{A.19})$$

617 Eq. (A.18) and Eq. (A.19) together give the desired result.

618 A.2 Proof of Proposition 2

619 Let $\mathcal{R} = \{\mathbf{x} \in \mathbb{S}^{d-1} | k(\mathbf{x}) = 0\}$ to be the region of the sphere that has agreement between the
620 unlearned and re-trained models. By assumption, we have $\text{surf}(\mathcal{R})/\text{surf}(\mathbb{S}^{d-1}) \leq 1/2$. Accordingly,
621 we define $\bar{\mathcal{R}} = \{\mathbf{x} \in \mathbb{S}^{d-1} | k(\mathbf{x}) = 1\}$ to be the complement of \mathcal{R} , denoting the region of the sphere
622 that has disagreement between the unlearned and re-trained models. Since $\text{surf}(\bar{\mathcal{R}})/\text{surf}(\mathbb{S}^{d-1}) \geq$
623 $1/2$, its τ -expansion $\mathcal{E}(\bar{\mathcal{R}}, p, \tau)$, by Lemma A.2, is at least as large as the epsilon expansion of a half
624 sphere; that is

$$\frac{\text{surf}(\mathcal{E}(\bar{\mathcal{R}}, p, \tau))}{\text{surf}(\mathbb{S}^{d-1})} \geq 1 - \left(\frac{\pi}{8}\right)^{1/2} \exp\left(-\frac{d-1}{2}\tau^2\right). \quad (\text{A.20})$$

625 Note that the τ -expansion $\mathcal{E}(\bar{\mathcal{R}}, p, \tau)$ represents the set of samples that either has $k(\mathbf{x}) = 1$ or admits
626 a $\mathbf{x}' \in \mathcal{B}_p(\mathbf{x}, \tau)$ such that $k(\mathbf{x}') = 1$. We can therefore define a set \mathcal{E}^c that is the complement of
627 $\mathcal{E}(\bar{\mathcal{R}}, p, \tau)$ with surface area satisfies

$$\frac{\text{surf}(\mathcal{E}^c)}{\text{surf}(\mathbb{S}^{d-1})} \leq \left(\frac{\pi}{8}\right)^{1/2} \exp\left(-\frac{d-1}{2}\tau^2\right). \quad (\text{A.21})$$

628 The probability to draw a sample in \mathcal{E}^c is then upper bounded by

$$\sup_{\mathbf{x}} \Pr[M(\mathbf{x}) = A(\mathbf{x})] \times \left(\frac{\pi}{8}\right)^{1/2} \exp\left(-\frac{d-1}{2}\tau^2\right). \quad (\text{A.22})$$

629 Using Lemma A.3, we know that with probability $2\delta/(1 - e^{-\epsilon})$,

$$\sup_{\mathbf{x}} \Pr[M(\mathbf{x}) = A(\mathbf{x})] = \sup_{\mathbf{x}} 1 - \Pr[M(\mathbf{x}) \neq A(\mathbf{x})] = 1 - (1 - \mathcal{O}(e^{-2\epsilon})) = \mathcal{O}(e^{-2\epsilon}). \quad (\text{A.23})$$

630 Putting Eq. (A.22) and Eq. (A.23) together, we know that the probability to draw a sample in
631 $\mathcal{E}(\bar{\mathcal{R}}, p, \tau)$ is at least

$$\begin{aligned} &\left(\frac{2\delta}{1 - e^{-\epsilon}}\right) \times \left\{1 - \sup_{\mathbf{x}} \Pr[M(\mathbf{x}) = A(\mathbf{x})] \times \left(\frac{\pi}{8}\right)^{1/2} \exp\left(-\frac{d-1}{2}\tau^2\right)\right\} \\ &= \left(\frac{2\delta}{1 - e^{-\epsilon}}\right) \times \left\{1 - \mathcal{O}(e^{-2\epsilon}) \times \left(\frac{\pi}{8}\right)^{1/2} \exp\left(-\frac{d-1}{2}\tau^2\right)\right\} \\ &= \left(\frac{2\delta}{1 - e^{-\epsilon}}\right) \times \left\{1 - \mathcal{O}\left[\left(\frac{\pi}{8}\right)^{1/2} \exp\left(-2\epsilon - \frac{d-1}{2}\tau^2\right)\right]\right\}. \end{aligned} \quad (\text{A.24})$$

B Details on the experimental setup

We provide implementation details of RURK in § 4.2, and introduce the unlearning baselines in § 5, including their mathematical formulations, operational interpretations, implementation specifics, and GitHub links. Finally, we summarize the dataset descriptions, training setups, and evaluation metrics.

B.1 Details of RURK in § 4.2

We first provide the pseudo-code of RURK in the following algorithm box Algorithm 1.

Algorithm 1: RURK Implementation

```

input      :Original  $\mathbf{w} = A(\mathcal{S})$ ; Forget Set  $\mathcal{S}_f$ ; Retain Set  $\mathcal{S}_r$ ; Loss Function  $\ell$ .
output     :Unlearned Model  $\mathbf{w}_{\text{RURK}}$ 
parameter :Perturbation Norm  $\tau$ ; Regularization Strength  $\lambda_f, \lambda_a$ ; Sample Size  $v$ ; # Epoch  $N$ 

1 RLoader  $\leftarrow$  MakeDataLoader( $\mathcal{S}_r$ );
2 FLoader  $\leftarrow$  MakeDataLoader( $\mathcal{S}_f$ );
3  $\mathbf{w}_{\text{RURK}} \leftarrow \mathbf{w}$ ;
4 for  $epoch \leftarrow 1$  to  $N$  do
5   for  $batchIdx, (retainData, retainTargets)$  in RLoader do    // Scan over all Batches
6      $retainOutput \leftarrow h_{\mathbf{w}}(retainData)$ ;
7      $\text{RLoss} \leftarrow \text{Loss}(retainOutput, retainTargets)$ ;
8      $forgetData, forgetTarget = \text{nextIter}(\text{FLoader})$ ;
9      $forgetOutput \leftarrow h_{\mathbf{w}}(forgetData)$ ;
10     $\text{FLoss} \leftarrow \text{Loss}(forgetOutput, forgetTarget)$ ;
11     $vulnerableSet = []$ ;
12    for  $i \leftarrow 1$  to  $v$  do    // Construct the Vulnerable Set  $\mathcal{V}((\mathbf{x}, y), \tau)$ 
13       $vulnerableSet \leftarrow vulnerableSet + \text{findAdv}(forgetData, forgetTarget, \tau)$ ;
14    end
15     $advForgetOutput \leftarrow h_{\mathbf{w}}(vulnerableSet)$ ;
16     $\text{AdvFLoss} \leftarrow \text{Loss}(advForgetOutput, forgetTarget)$ ;
17     $\text{Loss} \leftarrow \text{RLoss} - \lambda_f \text{FLoss} - \lambda_a \text{AdvFLoss}$ ;    // Entire Loss in Eq. (6)
18     $\mathbf{w}_{\text{RURK}} \leftarrow \mathbf{w}_{\text{RURK}} + \nabla_{\mathbf{w}_{\text{RURK}}} \text{Loss}$ ;    // SGD Optimization
19  end
20 end
return      : $\mathbf{w}_{\text{RURK}}$ 

```

We implement RURK using PyTorch (torch) (Paszke, 2019), and ensure reproducibility by fixing three random seeds: [131, 42, 7]. During optimization, we use a batch size of 128, the standard cross-entropy loss (`torch.nn.CrossEntropyLoss()`), and the SGD optimizer (`torch.optim.SGD`) with a learning rate of 0.01, momentum of 0.90, and weight decay of 5×10^{-4} . To stabilize training, we apply a cosine annealing learning rate scheduler (`torch.optim.lr_scheduler.CosineAnnealingLR`) and cap the total number of iterations at 200. Additionally, we clip the gradient norm to 1.0 using `torch.nn.utils.clip_grad_norm_`.

For both unlearning scenarios (small CIFAR-5 and CIFAR-10), we set the perturbation budget $\tau = 0.03$ and use the TorchAttacks library (Kim, 2020) to identify the vulnerable set $\mathcal{V}((\mathbf{x}, y), \tau)$. To improve efficiency, we define $\mathcal{V}((\mathbf{x}, y), \tau)$ as the entire perturbation ball $\mathcal{B}_p(\mathbf{x}, \tau)$ and set $v = 1$, meaning that only a single perturbed sample is drawn from a multivariate Gaussian distribution centered at \mathbf{x} with standard deviation τ .

We configure the hyper-parameters as follows: for small CIFAR-5, we use $N = 2$ and $\lambda_f = \lambda_a = 0.03$; for CIFAR-10, we set $N = 2$, $\lambda_f = 0.03$, and $\lambda_a = 0.00045$. Since $v = 1$ and the perturbation is generated via Gaussian noise, the inner loop in line 12 of Algorithm 1 executes only once, making the operations in lines 11–15 constant-time. As a result, the computational complexity of RURK is comparable to fine-tuning-based methods such as GA, GD, NegGrad+, and NGD.

However, if stronger adversaries like FGSM or PGD are used to identify the vulnerable set, the computational complexity increases from $\mathcal{O}(N)$ to $\mathcal{O}(N \times v)$. Note that unlike NGD, we do not inject additional noise into the gradient update steps.

658 **More discussion on RURK.** The regularization over the forget set in the loss $L(\mathbf{w}, A(\mathcal{S}))$ highlights
 659 two key features of our proposed unlearning method, RURK. First, under mild conditions, the un-
 660 learning procedure satisfies certified unlearning via Rényi Differential Privacy (RDP), providing
 661 formal guarantees—a certificate of unlearning. Second, enhancing robustness against vulnerable
 662 perturbations $\mathcal{V}((\mathbf{x}, y), \tau)$ may increase the distinguishability of the unlearned model from the ideal
 663 re-trained model.

664 To support the first point, consider the case where the forget set contains a single sample (i.e.,
 665 $|\mathcal{S}_f| = 1$), so that the original dataset \mathcal{S} and the retain set \mathcal{S}_r are adjacent. Assume the original model
 666 m was trained to satisfy (α, ϵ_0) -RDP. Further, suppose the unlearning procedure minimizes the loss
 667 in Eq. (6) using Projected Noisy Gradient Descent (PNGD). As shown by Chien et al. (2024), the
 668 Markov chain defined by PNGD updates of the form

$$\mathbf{w}_{t+1} = \Pi_R \left(\mathbf{w}_t - \eta \nabla L(\mathbf{w}_t, A(\mathcal{S})) + \sqrt{2\eta\sigma^2} W_t \right), \text{ with } \mathbf{w}_1 = A(\mathcal{S}) \quad (\text{B.25})$$

669 where $W_t \sim \mathcal{N}(0, \mathbb{I}_d)$ and Π_R denotes projection onto the ball of radius R , converges to a stationary
 670 distribution if the loss $L(\mathbf{w}, A(\mathcal{S}))$ from Eq. (6) is continuous. The first term of Eq. (6) is continuous
 671 by standard assumptions, as the loss $\ell(\cdot, (\mathbf{x}, y))$ is typically continuous in the weights. To show
 672 continuity of the second term, consider $g(\mathbf{w}) = \min_{\mathbf{z} \in \mathcal{B}_p(\mathbf{x}, \tau)} \ell(\mathbf{w}, (\mathbf{z}, y))$. Let \mathbf{z}^* be a minimizer;
 673 then $g(\mathbf{w}) = \ell(\mathbf{w}, (\mathbf{z}^*, y))$, which is continuous in \mathbf{w} since ℓ is. Therefore, the overall loss is
 674 continuous, and the PNGD process converges.

675 Furthermore, when the loss is L -smooth and M -Lipschitz, the stationary distribution obtained via
 676 PNGD satisfies (α, ϵ) -RDP for some ϵ depending on L , M , and convexity properties of the loss
 677 (Chien et al., 2024). For forget sets with multiple elements, one can sequentially apply this procedure,
 678 preserving Rényi unlearning at each step.

679 To justify the second point, observe that Term (ii) in Eq. (6) impacts the loss landscape’s smoothness.
 680 Specifically, the term $\kappa(\mathbf{w}, (\mathbf{x}, y))$ tends to increase with the perturbation radius τ , as the adversarial
 681 loss over a larger ball is generally higher. This leads to a gradient norm that increases with τ ,
 682 implying that the smoothness and Lipschitz constants of the overall loss also grow with τ . According
 683 to Theorem 3.2 in Chien et al. (2024), the Rényi distinguishability parameter ϵ is inversely related
 684 to these constants, implying a trade-off: achieving higher robustness (larger τ) increases model
 685 distinguishability (larger ϵ).

686 The assumptions underlying these results are mild and standard in the literature. We assume the loss
 687 is L -smooth and M -Lipschitz, and that the learning dynamics for the original dataset satisfy the
 688 Log-Sobolev Inequality (LSI) with constant C_{LSI} (Gross, 1975). Moreover, the Original $A(\mathcal{S})$ can
 689 be trained to satisfy (α, ϵ_0) -RDP under LSI using the same PNGD framework, as discussed by Chien
 690 et al. (2024).

691 B.2 Existing unlearning algorithms

692 Here, we provide detailed descriptions of the 11 unlearning baseline methods used in § 5, along with
693 links to their corresponding GitHub repositories.

694 **Certified Removal (CR).** Guo et al. (2020) propose a single-step Newton–Raphson update to remove
695 the influence of forget samples. Assuming that the empirical risk $L(\mathbf{w}, \mathcal{S})$ is twice differentiable with
696 continuous second derivatives, and letting $\mathbf{w}^* = \arg \min_{\mathbf{w} \in \mathcal{W}} L(\mathbf{w}, \mathcal{S}_r)$ denote the empirical risk
697 minimizer over the retain set \mathcal{S}_r , the Taylor expansion of the gradient ∇L around \mathbf{w}^* yields:

$$\nabla L(\mathbf{w}, \mathcal{S}_r) \approx \nabla L(\mathbf{w}_o, \mathcal{S}_r) + H_{\mathbf{w}_o}(\mathbf{w} - \mathbf{w}_o), \quad (\text{B.26})$$

698 where $\mathbf{w}_o = A(\mathcal{S})$ denotes the Original model parameters, and $H_{\mathbf{w}_o} = \nabla^2 L(\mathbf{w}_o, \mathcal{S}_r)$ is the
699 Hessian of the empirical risk over \mathcal{S}_r evaluated at \mathbf{w}_o . Re-arranging the terms gives:

$$\mathbf{w}^* \approx \mathbf{w}_o - H_{\mathbf{w}_o}^{-1} \nabla L(\mathbf{w}_o, \mathcal{S}_r), \quad (\text{B.27})$$

700 since $\nabla L(\mathbf{w}^*, \mathcal{S}_r) = 0$ by optimality. The certified removal method (CR) then defines the unlearned
701 model as

$$M(\mathbf{w}_o, \mathcal{S}, \mathcal{S}_f) = \mathbf{w}_o - \lambda H_{\mathbf{w}_o}^{-1} \nabla L(\mathbf{w}_o, \mathcal{S}_r), \quad (\text{B.28})$$

702 where λ is a step-size parameter. Under the assumption of convexity (ensuring the uniqueness of
703 minimizers) and bounded approximation error, Guo et al. (2020, Theorem 1) show that this procedure
704 guarantees (ϵ, δ) -certified removal, i.e., $M(\mathbf{w}_o, \mathcal{S}, \mathcal{S}_f) \stackrel{\epsilon, \delta}{\approx} \mathbf{w}^*$.

705 In our implementation, we follow the official codebase at [https://github.com/](https://github.com/facebookresearch/certified-removal)
706 [facebookresearch/certified-removal](https://github.com/facebookresearch/certified-removal). Since CR is designed for binary linear models,
707 we use a pre-trained ResNet-18 as a non-private feature extractor (excluding its final linear layer),
708 and apply the Newton update only to the final layer. Specifically, we train $|\mathcal{Y}|$ one-vs-rest logistic
709 regression classifiers on the extracted features and set $\lambda = 0.1$.

710 **Fisher Unlearning (Fisher).** Fisher (Golatk et al., 2020a) is one of the simplest unlearning
711 mechanisms, which removes the influence of forget samples by directly perturbing the model weights
712 with Gaussian noise (cf. §2.1). Unlike standard Gaussian perturbation, the variance of the noise is
713 scaled according to the inverse of the Fisher information matrix. Specifically, the unlearned model is
714 defined as:

$$M(\mathbf{w}, \mathcal{S}, \mathcal{S}_f) = \mathbf{w} + n, \quad n \sim \mathcal{N}(0, \alpha \mathbf{B}^{-1}), \quad (\text{B.29})$$

715 where \mathbf{B} denotes the Hessian matrix $\nabla^2 L(\mathbf{w}, \mathcal{S}_r)$. However, computing the exact Hessian is
716 computationally intractable even for moderately sized neural networks, and the matrix may not
717 be positive definite. To address this, Golatk et al. (2020a) approximate the Hessian using
718 the Levenberg–Marquardt algorithm, yielding a semi-positive-definite matrix closely related to
719 the Fisher information matrix (Martens, 2020), which motivates the name of the method. In
720 our implementation, we follow the official codebase and adopt the same hyper-parameters as in
721 <https://github.com/AdityaGolatk/SelectiveForgetting>.

722 **Neural Tangent Kernel Unlearning (NTK).** NTK (Golatk et al., 2020b) extends the Newton update
723 idea from CR by applying the Neural Tangent Kernel (NTK) linearization of neural networks. The
724 NTK matrix between two datasets \mathcal{S}_1 and \mathcal{S}_2 is defined as:

$$K(\mathcal{S}_1, \mathcal{S}_2) \triangleq \nabla_{\mathbf{w}} h_{\mathbf{w}}(\mathcal{S}_1) \nabla_{\mathbf{w}} f_{\mathbf{w}}(\mathcal{S}_2)^\top, \quad (\text{B.30})$$

725 where $h_{\mathbf{w}}$ denotes the network output and $h_{\mathbf{w}}$ may optionally denote a scalar output function (e.g.,
726 pre-activation logits). Let $\mathbf{w}^* = \arg \min_{\mathbf{w} \in \mathcal{W}} L(\mathbf{w}, \mathcal{S})$ and $\mathbf{w}_r = \arg \min_{\mathbf{w} \in \mathcal{W}} L(\mathbf{w}, \mathcal{S}_r)$ be the
727 minimizers of the empirical risk over the full and retain sets, respectively. By linearizing the network
728 output around \mathbf{w}^* using the NTK approximation, NTK enables a closed-form update that shifts the
729 model weights from \mathbf{w}^* to an approximation of \mathbf{w}_r via a one-shot adjustment:

$$\mathbf{w}_r \approx M(\mathbf{w}^*, \mathcal{S}, \mathcal{S}_f) = \mathbf{w}^* + \mathbf{P} \nabla_{\mathbf{w}^*} h_{\mathbf{w}^*}(\mathcal{S}_f)^\top \mathbf{M} \mathbf{V}, \quad (\text{B.31})$$

where the projection matrix $\mathbf{P} = \mathbf{I} - \nabla_{\mathbf{w}^*} h_{\mathbf{w}^*}(\mathcal{S}_r)^\top K(\mathcal{S}_r, \mathcal{S}_r)^{-1} \nabla_{\mathbf{w}^*} h_{\mathbf{w}^*}(\mathcal{S}_r)$ ensures that the gradient contributions from the forget set are orthogonalized against those from the retain set. The matrix \mathbf{M} is defined as

$$\mathbf{M} = [K(\mathcal{S}_f, \mathcal{S}_f) - K(\mathcal{S}_r, \mathcal{S}_f)^\top K(\mathcal{S}_r, \mathcal{S}_r)^{-1} K(\mathcal{S}_r, \mathcal{S}_f)]^{-1}, \quad (\text{B.32})$$

and the re-weighting matrix \mathbf{V} is given by

$$\mathbf{V} = (y_f - h_{\mathbf{w}^*}(\mathcal{S}_f)) + K(\mathcal{S}_r, \mathcal{S}_f)^\top K(\mathcal{S}_f, \mathcal{S}_f)^{-1} (y_r - h_{\mathbf{w}^*}(\mathcal{S}_r)), \quad (\text{B.33})$$

where y_f and y_r denote the ground truth labels for the forget and retain sets, respectively. We follow the official implementation and use the same settings and hyper-parameters as in <https://github.com/AdityaGolatkhar/SelectiveForgetting>.

Gradient Descent (GD). GD (Neel et al., 2021) is one of the simplest unlearning algorithms. It continues training the original model `Original` on the retain set \mathcal{S}_r using standard gradient descent. Specifically, the unlearned model M is obtained by iteratively applying the update:

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta g_t(\mathbf{w}_t), \quad \text{with} \quad \mathbf{w}_1 = A(\mathcal{S}),$$

where η is the step size and $g_t(\mathbf{w}_t)$ is a (mini-batch) stochastic gradient of the empirical loss over \mathcal{S}_r , i.e., $\frac{1}{|\mathcal{S}_r|} \sum_{\mathbf{s} \in \mathcal{S}_r} \ell(\mathbf{w}, \mathbf{s})$. The key intuition behind GD is that when the forget set is small (i.e., $|\mathcal{S}_f| \ll |\mathcal{S}|$), the minimizers of the loss functions over \mathcal{S} and \mathcal{S}_r are expected to be close. Therefore, a few steps of gradient descent starting from the original model $\mathbf{w}_1 = A(\mathcal{S})$ can efficiently move the parameters toward a minimizer of the updated training objective. Building on this intuition, Neel et al. (2021) also provide theoretical guarantees for the effectiveness of GD in both convex and certain non-convex settings.

Our implementation follows <https://github.com/ChrisWaites/descent-to-delete>, using the hyper-parameters :

- SGD optimizer with a lr = 1e-2, momentum = 0.9, and weight decay = 1e-4.
- Random seed for the data loader is 7; random seeds for repeated experiments are 131, 42, 7.
- Batch size = 128
- Number of epochs = 10

Noisy Gradient Descent (NGD). NGD (Chourasia and Shah, 2023) is a simple extension of GD in which Gaussian noise is added to each gradient update. The unlearned model is obtained by iteratively applying the update:

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta (g_t(\mathbf{w}_t) + \xi_t), \quad \text{with} \quad \mathbf{w}_1 = A(\mathcal{S}),$$

where η is the step size, $g_t(\mathbf{w}_t)$ denotes a (mini-batch) stochastic gradient of the empirical loss over the retain set \mathcal{S}_r , i.e., $\frac{1}{|\mathcal{S}_r|} \sum_{\mathbf{s} \in \mathcal{S}_r} \ell(\mathbf{w}, \mathbf{s})$, and $\xi_t \sim \mathcal{N}(0, \sigma^2)$ is an independent Gaussian noise term added at each iteration. The key distinction between NGD and GD lies in the injection of noise during optimization. This stochasticity not only increases the robustness of the unlearning process but also enables formal privacy guarantees, as demonstrated in the context of certified unlearning via Rényi differential privacy (Chien et al., 2024). Notably, a similar noise-injection mechanism is employed in the DP-SGD algorithm for training models with differential privacy guarantees (Abadi et al., 2016).

Our implementation follows https://github.com/Graph-COM/Langevin_unlearning and the same hyper-parameters as GD with

- $(\sigma, \eta) = (0.03, 0.1)$, trained for 10 epochs for the small CIFAR-5 settings with sample unlearning.
- $(\sigma, \eta) = (0.03, 0.1)$, trained for 2 epochs for the CIFAR-10 settings with sample unlearning.

Gradient Descent (GA). GA (Graves et al., 2021) aims to remove the influence of the forget set \mathcal{S}_f from a trained model by reversing the learning process through gradient ascent. It stores all gradient updates involving \mathcal{S}_f during the initial training phase, and unlearning is then performed by applying the exact reverse updates—i.e., gradient ascent—using the stored gradients. However, this

approach is highly memory-intensive and becomes impractical for large-scale models. To address this limitation, Jang et al. (2022) proposed a more scalable variant, in which unlearning is achieved by performing mini-batch gradient ascent on the forget set. Specifically, the model is updated by minimizing the negated loss $-L(\mathbf{w}, \mathcal{S}_f)$, effectively implementing ascent steps that counteract the original training influence of \mathcal{S}_f .

Our implementation follows <https://github.com/joeljang/knowledge-unlearning>, using the similar hyper-parameters as GD:

- SGD optimizer with a lr = 1e-5, momentum = 0.9, and weight decay = 1e-4, clipping gradient norm = 1.
- Random seed for the data loader is 7; random seeds for repeated experiments are 131, 42, 7.
- Batch size = 128
- Number of epochs = 1 for Small-CIFAR-5 (both sample and class unlearning) and CIFAR-10 with sample unlearning; while 3 for CIFAR-10 with class unlearning

Negative Gradient Plus (NegGrad+). NegGrad+ (Kurmanji et al., 2024) fine-tunes the model by simultaneously minimizing the loss on the retain set \mathcal{S}_r and maximizing the loss on the forget set \mathcal{S}_f , effectively negating the gradient contributions from the latter. Specifically, the unlearned model is obtained by optimizing the following objective:

$$L(\mathbf{w}, \mathcal{S}_r) - \beta L(\mathbf{w}, \mathcal{S}_f), \quad (\text{B.34})$$

where β is a hyper-parameter that controls the strength of unlearning by weighting the loss contribution from \mathcal{S}_f . NegGrad+ shares conceptual similarity with the Gradient Ascent (GA) method, as both perform loss maximization on the forget set to induce forgetting. However, NegGrad+ is empirically more stable and yields better performance, as it simultaneously enforces loss minimization on the retain set, ensuring that useful knowledge is preserved during unlearning.

Our implementation follows <https://github.com/meghdadk/SCRUB>, using similar hyper-parameters as GD and GA:

- SGD optimizer with a lr = 1e-2, momentum = 0.9, and weight decay = 1e-4, clipping gradient norm = 1.
- $\beta = 0.001$
- Random seed for the data loader is 7; random seeds for repeated experiments are 131, 42, 7.
- Batch size = 128
- Number of epochs = 1 for Small-CIFAR-5 (both sample and class unlearning) and CIFAR-10 with sample unlearning; while 3 for CIFAR-10 with class unlearning

Exact Unlearning the last k layers (EU- k). EU- k (Goel et al., 2022) is a simple, parameter-efficient unlearning method designed for deep learning models, requiring access only to the retain set \mathcal{S}_r . Given a parameter k , EU- k retrains from scratch the last k layers of the neural network—those closest to the output layer—while keeping the earlier layers fixed. By adjusting k , EU- k provides a tunable trade-off between unlearning effectiveness and computational efficiency.

Our implementation is based on the official repository at <https://github.com/shash42/Evaluating-Inexact-Unlearning>, using the following hyper-parameters:

- SGD optimizer with a lr = 1e-2, momentum = 0.9, and weight decay = 1e-4.
- Last fully-connected layer, and last two residual blocks are reinitialized, and only fine-tune those parameters.
- Random seed for the data loader is 7; random seeds for repeated experiments are 131, 42, 7.
- Batch size = 128
- Number of epochs = 10

816 **Catastrophically Forgetting the last k layers (CF-k).** CF-k (Goel et al., 2022) builds on the
817 observation that neural networks tend to forget information about data samples encountered early in
818 training—a phenomenon known as catastrophic forgetting (French, 1999). Similar to EU-k, CF-k
819 focuses on the last k layers of the network; however, instead of re-training these layers from scratch,
820 it fine-tunes them starting from the original model parameters $\mathbf{w} = A(\mathcal{S})$, using only the retain set
821 \mathcal{S}_r , while keeping all earlier layers frozen.

822 As with EU-k, our implementation follows the official repository at <https://github.com/shash42/Evaluating-Inexact-Unlearning>, using the following hyper-parameters:

- 824 • SGD optimizer with a lr = 1e-2, momentum = 0.9, and weight decay = 1e-4.
- 825 • Last fully-connected layer, and last two residual blocks are fine-tuned.
- 826 • Random seed for the data loader is 7; random seeds for repeated experiments are 131, 42, 7.
- 827 • Batch size = 128
- 828 • Number of epochs = 10

829 **S**calable **R**emembering and **U**nlearning **u**nBound (SCRUB). SCRUB (Kurmanji et al., 2024) is
830 one of the state-of-the-art unlearning methods for deep learning. It formulates the unlearning task
831 within a student–teacher framework: given a trained teacher model \mathbf{w}_T , the goal is to train a student
832 model \mathbf{w} that selectively imitates the teacher. Specifically, the student should retain the teacher’s
833 behavior on the retain set \mathcal{S}_r while diverging significantly from it on the forget set \mathcal{S}_f , as measured
834 by the KL divergence. To achieve this, SCRUB optimizes a modified knowledge distillation objective
835 (Hinton et al., 2014):

$$\mathbb{E}_{(\mathbf{x}, y) \sim \mathcal{S}_r} [\alpha, D_{\text{KL}}(h_{\mathbf{w}_T}(\mathbf{x}) | h_{\mathbf{w}}(\mathbf{x})) + \gamma, \ell(\mathbf{w}, (\mathbf{x}, y))] - \mathbb{E}_{(\mathbf{x}, y) \sim \mathcal{S}_f} [D_{\text{KL}}(h_{\mathbf{w}_T}(\mathbf{x}) | h_{\mathbf{w}}(\mathbf{x}))], \quad (\text{B.35})$$

836 where α and γ are hyper-parameters that balance knowledge retention and task performance. The
837 first expectation encourages the student to match the teacher on the retain set while also performing
838 well on the classification task, and the second term enforces divergence from the teacher on the forget
839 set.

840 Our implementation follows <https://github.com/meghdadk/SCRUB>, with the hyper-parameters:

- 841 • SGD optimizer with a lr = 5e-4, momentum = 0.9, and weight decay = 5e-4.
- 842 • Random seed for the data loader is 7; random seeds for repeated experiments are 131, 42, 7.
- 843 • Batch size = 128
- 844 • $\alpha = 0.001$
- 845 • $\gamma = 1$ for small CIFAR-5 settings with both sample and class unlearning
- 846 • $\gamma = 0.6$ for CIFAR-10 setting with sample unlearning and $\gamma = 75$ for CIFAR-10 with class
847 unlearning

848 **S**elective **S**ynaptic **D**ampening (SSD). SSD was introduced by Foster et al. (2024) as a method
849 to unlearn a specific forget set from a neural network without retraining it from scratch. Building
850 on ideas similar to Fisher, but with a more refined approach, SSD selectively dampens weights
851 that exhibit disproportionately high influence—measured via the Fisher information—on the forget
852 set relative to the retain set. Given a model with parameters \mathbf{w} , let \mathbf{F}_r and \mathbf{F}_f denote the Fisher
853 information matrices computed over the retain set \mathcal{S}_r and the forget set \mathcal{S}_f , respectively. Unlearning
854 is performed by scaling each parameter \mathbf{w}_i according to its relative Fisher sensitivity:

$$\mathbf{w}_i = \begin{cases} \beta \mathbf{w}_i & \text{if } \mathbf{F}_{f,i} > \alpha \mathbf{F}_{r,i}, \\ \mathbf{w}_i & \text{otherwise,} \end{cases} \quad (\text{B.36})$$

855 where $\mathbf{F}_{f,i}$ and $\mathbf{F}_{r,i}$ denote the i -th diagonal entries of the Fisher matrices for \mathcal{S}_f and \mathcal{S}_r , respectively.
856 The hyper-parameter α controls the threshold for selecting influential weights, and the dampening
857 factor β is defined as $\beta = \min_i \lambda \mathbf{F}_{r,i} / \mathbf{F}_{f,i}, 1$ for a tunable parameter λ .

858 Our implementation is based on the official repository at <https://github.com/if-loops/selective-synaptic-dampening>, using the following hyper-parameters:

- 860 • $\lambda = 1.0$
- 861 • $\alpha = 10.0$ for CIFAR-10 and 100.0 for Small-CIFAR-5.

B.3 Training and evaluation details

Here, we provide details on dataset preparation, unlearning settings, and evaluation procedures.

Dataset and unlearning settings. We evaluate unlearning methods in the context of image classification using CIFAR-10 (Krizhevsky et al., 2009) and ResNet-18 (He et al., 2016), focusing on the random sample unlearning setting across two scenarios. The CIFAR-10 dataset contains 60,000 color images of size 32×32 pixels, evenly distributed across 10 classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. Each class includes 5,000 training and 1,000 test samples. CIFAR-10 is widely used in machine unlearning research—for example, by Kurmanji et al. (2024); Chien et al. (2024); Golatkar et al. (2020a,b); Foster et al. (2024). In the first scenario—small CIFAR-5—we follow the setup of Golatkar et al. (2020a,b) by creating a reduced version of CIFAR-10 with 200 training and 200 test samples from each of the first five classes. From class 0, we randomly select 100 samples (50%) as the forget set \mathcal{S}_f . The second scenario uses the full CIFAR-10 dataset. Here, we designate 2,000 samples (50%⁸) from class 0 as the forget set \mathcal{S}_f . In addition to sample unlearning, we also consider class unlearning. In this setting, the forget set contains 200 samples from class 0 in the small CIFAR-5 scenario, and 4,000 samples from class 0 in the full CIFAR-10 scenario. All experiments are conducted on an AWS EC2 g5.24xlarge instance.

Training of Original and Re-train. To avoid any external knowledge (e.g., the default ImageNet pre-trained weights from torchvision (Marcel and Rodriguez, 2010)), both the Original and Re-train models are trained entirely from scratch, without loading any pre-trained weights. We use a batch size of 128 and cross-entropy loss (`torch.nn.CrossEntropyLoss()`). For the small CIFAR-5 setting, we first pre-train a ResNet-18 model on the full CIFAR-10 dataset using the SGD optimizer (`torch.optim.SGD(lr=0.4, momentum=0.9, weight_decay=2e-4)`) along with a cosine annealing learning rate scheduler (`torch.optim.lr_scheduler.CosineAnnealingLR(T_max=200)`). We then fine-tune this model on small CIFAR-5 for 25 epochs with a learning rate of 0.01 to obtain the Original model reported in Table 1. The Re-train model in Table 1 is obtained using the same setup, except the fine-tuning is performed on the retain set \mathcal{S}_r for 100 epochs. For the CIFAR-10 setting, both the Original and Re-train models are trained from scratch on the full dataset and the retain set, respectively, for 137 epochs using a learning rate of 0.01. For the VGG-11 ablation study, we follow the same training protocol, including the optimizer, scheduler, and learning rate.

Evaluation details. The retain, forget, and test accuracies reported in Table 1 are computed by directly evaluating each model on the corresponding retain, forget, and test sets. The unlearning accuracy is defined as $1 - \text{forget accuracy}$. The re-learn time quantifies how easily a model can reacquire knowledge of the forget set. It is defined as the number of fine-tuning epochs required for the model m to satisfy $L(m, \mathcal{S}_f) \leq (1 + \eta)L(\text{Original}, \mathcal{S}_f)$, where we set $\eta = 0.05$. This value can be adjusted depending on the desired tolerance.

To evaluate vulnerability to membership inference attacks (MIA), we use a support vector classifier (SVC) implemented via the `scikit-learn` package, with parameters `SVC(C=3, gamma='auto', kernel='rbf')` (Pedregosa et al., 2011). The SVC is trained to distinguish between samples seen during training and those that were not, based on the model’s output likelihood for the correct class label. In the sample unlearning setting, retain samples are labeled as 1 (seen) and test samples as 0 (unseen) during SVC training; forget samples are then evaluated by the SVC, and an ideal unlearned model should cause them to be classified as 0. In the class unlearning setting, retain samples are labeled as 1 and forget samples as 0 for SVC training; evaluation is then conducted on a held-out subset of the forget class. The reported MIA Accuracy in Table 1 corresponds to the SVC’s attack failure rate, i.e., the proportion of forget samples classified as unseen.

Both the construction of the vulnerable set $\mathcal{V}((\mathbf{x}, y), \tau)$ and the evaluation of residual knowledge $r_\tau(\mathcal{S}_f)$ involve generating perturbed inputs within the norm ball $\mathcal{B}_p(\mathbf{x}, \tau)$. We implement this using the `torchattacks` package (Kim, 2020). For Gaussian noise (i.e., $p = 2$), we use `torchattacks.GN(model, std= τ)`; for FGSM (i.e., $p = \infty$), we use `torchattacks.FGSM(model, eps= τ)`; and for PGD, we use `torchattacks.PGD(model,`

⁸Each class has 5,000 training samples, with 20% held out for validation.

913 `eps= τ , alpha=2/255., steps=pgd_epoch, random_start=True)`. We apply targeted at-
 914 tacks for both FGSM and PGD.

915 To estimate residual knowledge, we apply these attacks to the unlearned model m to generate per-
 916 turbed inputs \mathbf{x}' , and compute both $\Pr[m(\mathbf{x}') = y]$ and $\Pr[a(\mathbf{x}') = y]$ over $c = 100$ independent runs,
 917 using the same perturbation \mathbf{x}' for both models. Constructing the vulnerable set $\mathcal{V}((\mathbf{x}, y), \tau)$ involves
 918 solving the minimization problem $\min_{\mathbf{z} \in \mathcal{B}_p(\mathbf{x}, \tau)} \ell(\mathbf{w}, (\mathbf{z}, y))$. However, since the `torchattacks`
 919 package is designed to solve the adversarial objective $\max_{\mathbf{z} \in \mathcal{B}_p(\mathbf{x}, \tau)} \ell(\mathbf{w}, (\mathbf{z}, y))$, we instead define
 920 a random target label $y' \neq y$ and solve $\max_{\mathbf{z} \in \mathcal{B}_p(\mathbf{x}, \tau)} \ell(\mathbf{w}, (\mathbf{z}, y'))$. To adapt this into our residual
 921 knowledge framework, we flip the sign of the regularization term in Eq. (6), effectively encouraging
 922 the model to associate perturbed variants of forget samples with an incorrect label y' . This discour-
 923 ages the model from retaining knowledge of the true label and facilitates unlearning. We repeat this
 924 process v times for each forget sample to construct its corresponding vulnerable set.

C Additional results and experiments

We include additional experimental results and comparisons, including: (i) comprehensive results on sample unlearning for both the small CIFAR-5 and CIFAR-10 scenarios, covering accuracy metrics and residual knowledge estimates under various adversarial attacks; (ii) results on class unlearning for both small CIFAR-5 and CIFAR-10; (iii) ablation studies on the hyper-parameters of RURK; and (iv) an analysis of the prevalence of residual knowledge across different settings.

C.1 Complete results on sample unlearning for small CIFAR-5

We present the full version of Table 1 under the small CIFAR-5 setting in Table C.2. Combined with Table 1, the results demonstrate that RURK consistently achieves superior performance—surpassing certified removal methods such as CR, Fisher, and NTK, as well as deep-learning-compatible approaches like SCRUB, NegGrad+, and NGD—by achieving the smallest average gap from the re-trained model Re-train. Notably, the re-learn time of RURK is also comparable to that of Re-train.

We further include the complete residual knowledge curves estimated under different adversarial attacks: Gaussian noise in Figure C.3, FGSM in Figure C.4, and PGD in Figure C.5. For both Gaussian noise and FGSM, most methods exhibit residual knowledge greater than 1—i.e., inheriting information from the Original model—whereas RURK maintains residual knowledge close to 1 for small perturbation norms τ , and reduces it below 1 as τ increases.

With PGD (10 steps), a stronger attack, we observe that methods such as NTK, NGD, and SCRUB still suffer from residual knowledge. In contrast, methods like GD, GA, NegGrad+, EU-k, CF-k, and SSD maintain residual knowledge near 1 across varying values of τ . In particular, SSD strictly preserves residual knowledge at 1, though this comes at the cost of a larger average gap from Re-train (cf. Table C.2).

Overall, RURK effectively suppresses residual knowledge to values below 1 for small τ , and keeps it close to 1 as τ increases—striking a favorable balance between preserving accuracy and mitigating residual knowledge.

Table C.2: The complete performance summary of various unlearning methods on small CIFAR-5 (cf. Table 1). Results are reported in the format $a \pm b$, indicating the mean a and standard deviation b over 3 independent trials. The absolute performance gap relative to Re-train is shown in (blue). For methods that fail to recover the forget-set knowledge within 30 training epochs, the re-learn time is reported as “>30”.

Datasets	Methods	Evaluation Metrics					
		Retain Acc. (%)	Unlearn Acc. (%)	Test Acc. (%)	MIA Acc. (%)	Avg. Gap	Re-learn Time (# Epoch)
Small CIFAR-5	Original	99.93 \pm 0.10(0.03)	0.00 \pm 0.00(8.33)	95.37 \pm 0.80(0.57)	4.67 \pm 3.30(22.33)	7.82	-
	Re-train	99.96 \pm 0.05(0.00)	8.33 \pm 3.30(0.00)	94.80 \pm 0.85(0.00)	27.00 \pm 5.66(0.00)	0.00	3.33 \pm 0.47
	CR	99.56 \pm 0.47(0.40)	14.00 \pm 5.66(5.67)	91.80 \pm 0.99(3.00)	58.17 \pm 0.79(31.17)	10.06	-
	Fisher	92.67 \pm 0.63(7.29)	12.67 \pm 0.94(4.34)	88.80 \pm 1.98(6.00)	47.33 \pm 6.13(20.33)	9.49	3.00 \pm 1.41
	NTK	99.93 \pm 0.10(0.03)	7.00 \pm 0.00(1.33)	95.37 \pm 0.80(0.57)	16.00 \pm 4.24(11.00)	3.23	4.67 \pm 0.47
	GD	84.04 \pm 0.42(15.93)	22.67 \pm 5.19(14.33)	76.83 \pm 1.04(17.97)	84.67 \pm 10.84(27.33)	18.89	12.67 \pm 12.97
	NGD	95.07 \pm 1.36(4.89)	4.67 \pm 0.47(3.66)	89.33 \pm 2.03(5.47)	13.33 \pm 2.36(13.67)	6.92	0.67 \pm 0.47
	GA	94.22 \pm 2.83(5.74)	40.67 \pm 5.19(32.33)	86.37 \pm 0.24(8.43)	84.00 \pm 11.31(26.67)	18.29	2.33 \pm 0.47
	NegGrad+	96.78 \pm 2.04(3.19)	25.00 \pm 1.41(16.67)	89.77 \pm 0.66(5.03)	74.67 \pm 17.91(17.33)	10.55	2.00 \pm 0.00
	EU-k	91.15 \pm 5.92(8.81)	20.00 \pm 4.24(11.67)	76.33 \pm 4.05(18.47)	37.00 \pm 4.24(20.33)	14.82	9.33 \pm 6.85
	CF-k	99.96 \pm 0.05(0.00)	0.33 \pm 0.47(8.00)	94.73 \pm 0.75(0.07)	37.00 \pm 24.04(20.33)	7.10	17.00 \pm 11.43
	SCRUB	99.88 \pm 0.18(0.08)	1.33 \pm 0.47(7.00)	94.67 \pm 0.79(0.13)	18.33 \pm 9.43(8.67)	3.97	1.00 \pm 0.00
	SSD	96.85 \pm 1.20(3.11)	13.33 \pm 8.01(5.00)	89.43 \pm 2.88(5.37)	69.00 \pm 8.49(11.67)	6.29	2.33 \pm 0.47
	RURK	99.52 \pm 0.37(0.44)	5.67 \pm 2.36(2.66)	93.83 \pm 0.90(0.97)	33.33 \pm 12.26(6.33)	2.60	2.00 \pm 0.00

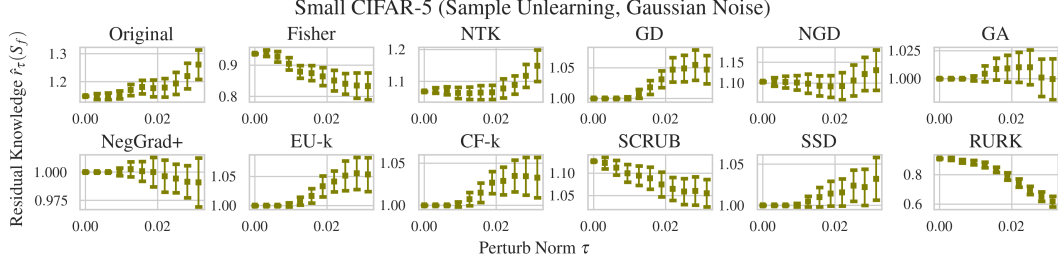


Figure C.3: Residual knowledge $\hat{r}_\tau(\mathcal{S}_f)$ of the proposed RURK, Original, and other unlearning methods on small CIFAR-5 with sample unlearning, evaluated under varying perturbation norms τ , using Gaussian noise ($p = 2$) to draw $c = 100$ samples from $\mathcal{B}_p(\mathbf{x}, \tau)$.

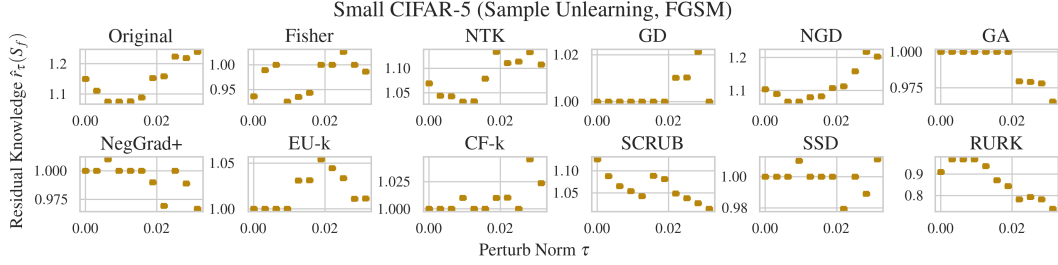


Figure C.4: Residual knowledge $\hat{r}_\tau(\mathcal{S}_f)$ of the proposed RURK, Original, and other unlearning methods on small CIFAR-5 with sample unlearning, evaluated under varying perturbation norms τ , using targeted FGSM ($p = \infty$) to draw $c = 100$ samples from $\mathcal{B}_p(\mathbf{x}, \tau)$.

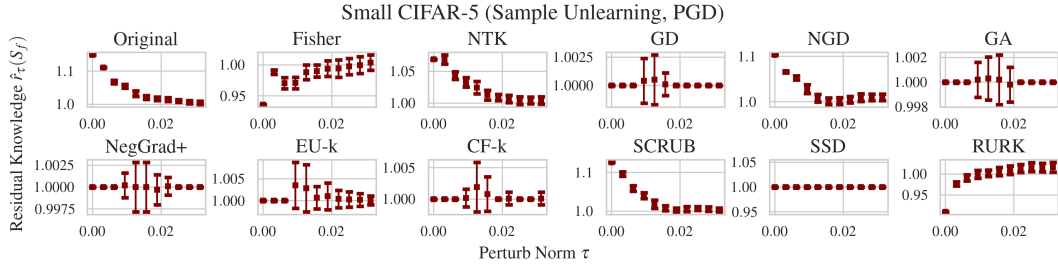


Figure C.5: Residual knowledge $\hat{r}_\tau(\mathcal{S}_f)$ of the proposed RURK, Original, and other unlearning methods on small CIFAR-5 with sample unlearning, evaluated under varying perturbation norms τ , using targeted PGD ($p = \infty$) to draw $c = 100$ samples from $\mathcal{B}_p(\mathbf{x}, \tau)$.

950 We provide the residual knowledge estimates under FGSM in Figure C.6 and under PGD in Figure C.7,
 951 corresponding to the results presented in Table 1 and Figure 2 of the main text. RURK demonstrates
 952 consistent behavior across all adversarial attack types—Gaussian noise, FGSM, and PGD—by
 953 maintaining residual knowledge close to 1 for small perturbation radii τ , and effectively suppressing
 954 it below 1 as τ increases.

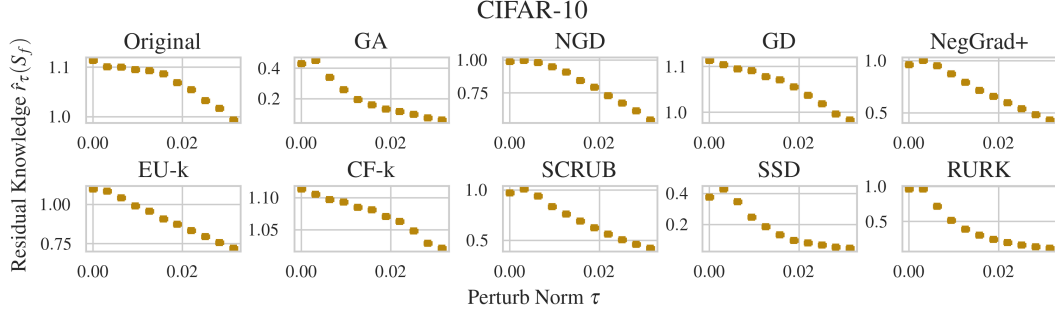


Figure C.6: Residual knowledge $\hat{r}_\tau(S_f)$ of the proposed RURK, Original, and other unlearning methods on CIFAR-10 with sample unlearning, evaluated under varying perturbation norms τ , using targeted FGSM ($p = \infty$) to draw $c = 100$ samples from $\mathcal{B}_p(\mathbf{x}, \tau)$.

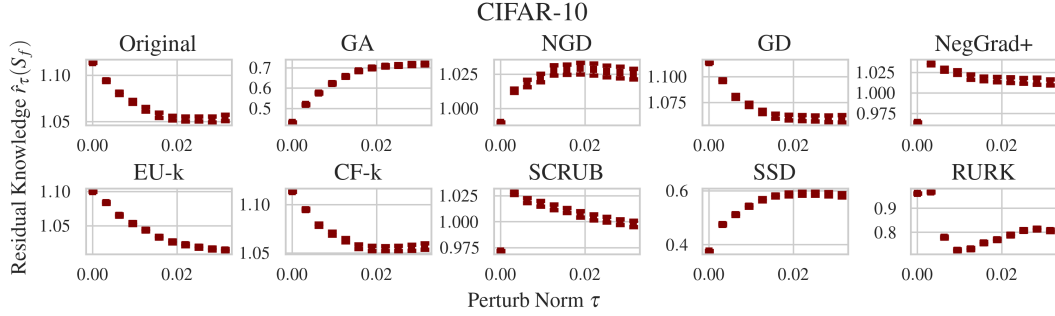


Figure C.7: Residual knowledge $\hat{r}_\tau(S_f)$ of the proposed RURK, Original, and other unlearning methods on CIFAR-10 with sample unlearning, evaluated under varying perturbation norms τ , using targeted PGD ($p = \infty$) to draw $c = 100$ samples from $\mathcal{B}_p(\mathbf{x}, \tau)$.

C.2 Class unlearning: Forgetting all samples in a class

In the main text, we focus on the sample unlearning setting, where 50% of the samples from class 0 are unlearned. Here, we provide analogous results for the class unlearning setting, where 100% of the samples from class 0 are removed, on both small CIFAR-5 and CIFAR-10. Note that in class unlearning, the Re-train model is never exposed to any samples from class 0. As a result, the unlearning accuracy is 100%—that is, Re-train never correctly classifies any forget sample into class 0. In Table C.3, we report the same evaluation metrics as in Table 1, including the average absolute gaps in retain, unlearn, test, and MIA accuracy compared to Re-train.

In both small CIFAR-5 and CIFAR-10, EU-k achieves the smallest average gap relative to Re-train. This result is expected, as class unlearning in this case resembles a transfer learning scenario: from a source domain (small CIFAR-5/CIFAR-10) to a target domain (small CIFAR-4/CIFAR-9), where the forget class is entirely absent. This creates a clear domain shift, allowing EU-k to perform well by fine-tuning only the top layers. In contrast, in sample unlearning, both the source and target domains still contain samples from class 0, making transfer learning less effective and leading to poorer performance for EU-k.

We also present residual knowledge estimates for the small CIFAR-5 class unlearning scenario under Gaussian noise (Figure C.8), FGSM (Figure C.9), and PGD (Figure C.10). While RURK is the second-best performer in terms of average accuracy gap in the small CIFAR-5 setting, it outperforms EU-k in suppressing residual knowledge. In the CIFAR-10 class unlearning scenario, RURK is the best among popular methods such as GA, SCRUB and SSD.

Table C.3: Performance summary of various unlearning methods for class unlearning. Results are reported in the format $a \pm b$, indicating the mean a and standard deviation b over 3 independent trials. The absolute performance gap relative to Re-train is shown in (blue). For methods that fail to recover the forget-set knowledge within 30 training epochs, the re-learn time is reported as “>30”.

Datasets	Methods	Evaluation Metrics					
		Retain Acc. (%)	Unlearn Acc. (%)	Test Acc. (%)	MIA Acc. (%)	Avg. Gap	Re-learn Time (# Epoch)
Small CIFAR-5	Original	99.92 \pm 0.12(0.30)	0.00 \pm 0.00(100.00)	95.25 \pm 0.88(1.75)	26.50 \pm 4.95(73.50)	43.89	-
	Re-train	99.79 \pm 0.12(0.17)	100.00 \pm 0.00(0.00)	93.33 \pm 0.12(0.00)	100.00 \pm 0.00(0.00)	0.00	1.00 \pm 0.00
	Fisher	93.75 \pm 0.71(6.04)	14.33 \pm 0.47(85.67)	89.12 \pm 2.12(4.20)	54.67 \pm 6.84(45.33)	35.31	3.33 \pm 0.47
	NTK	25.00 \pm 0.00(74.79)	100.00 \pm 0.00(0.00)	25.00 \pm 0.00(68.33)	100.00 \pm 0.00(0.00)	35.78	9.67 \pm 0.47
	GD	67.29 \pm 0.41(32.50)	93.67 \pm 0.24(6.33)	62.75 \pm 2.65(30.58)	89.83 \pm 6.60(10.17)	19.90	31.00 \pm 0.00
	NGD	91.38 \pm 0.00(8.42)	100.00 \pm 0.00(0.00)	59.75 \pm 0.00(33.58)	100.00 \pm 0.00(0.00)	10.50	12.33 \pm 1.89
	GA	98.17 \pm 1.71(1.62)	33.00 \pm 5.66(67.00)	92.79 \pm 0.94(0.54)	64.33 \pm 0.94(35.67)	26.21	2.00 \pm 0.00
	NegGrad+	99.50 \pm 0.53(0.29)	19.83 \pm 5.42(80.17)	93.83 \pm 0.65(0.50)	58.67 \pm 2.59(41.33)	30.57	2.00 \pm 0.00
	EU-k	99.46 \pm 0.06(0.33)	100.00 \pm 0.00(0.00)	84.50 \pm 0.35(8.83)	100.00 \pm 0.00(0.00)	2.29	31.00 \pm 0.00
	CF-k	99.96 \pm 0.06(0.17)	42.33 \pm 9.66(57.67)	95.79 \pm 0.41(2.46)	97.83 \pm 2.36(2.17)	15.61	31.00 \pm 0.00
	SCRUB	99.96 \pm 0.06(0.17)	7.33 \pm 5.42(92.67)	95.5 \pm 0.35(2.2)	43.67 \pm 1.18(56.33)	37.84	1.00 \pm 0.00
	SSD	98.46 \pm 0.24(1.33)	7.00 \pm 9.90(93.00)	93.29 \pm 0.77(0.04)	42.17 \pm 4.48(57.83)	38.05	2.00 \pm 0.00
	RURK	98.58 \pm 1.65(1.21)	87.17 \pm 2.59(12.83)	93.71 \pm 0.65(0.38)	96.33 \pm 5.19(3.67)	4.52	1.33 \pm 0.47
CIFAR-10	Original	100.00 \pm 0.00(0.00)	0.00 \pm 0.00(100.00)	94.71 \pm 0.06(0.82)	7.87 \pm 0.29(92.13)	48.24	-
	Re-train	100.00 \pm 0.00(0.00)	100.00 \pm 0.00(0.00)	93.89 \pm 0.14(0.00)	100.00 \pm 0.00(0.00)	0.00	12.67 \pm 4.64
	GD	99.98 \pm 0.02(0.02)	0.09 \pm 0.06(99.91)	94.33 \pm 0.09(0.44)	29.27 \pm 0.79(70.73)	42.78	0.67 \pm 0.47
	NGD	99.91 \pm 0.01(0.09)	20.84 \pm 0.35(79.16)	94.40 \pm 0.02(0.51)	96.43 \pm 0.12(3.57)	20.83	8.00 \pm 3.27
	GA	95.51 \pm 0.16(4.49)	93.60 \pm 0.07(6.40)	88.60 \pm 0.21(5.29)	96.47 \pm 0.31(3.53)	4.93	1.00 \pm 0.82
	NegGrad+	99.93 \pm 0.00(0.07)	57.19 \pm 0.58(42.81)	94.30 \pm 0.06(0.41)	86.80 \pm 0.37(13.20)	14.12	0.67 \pm 0.47
	EU-k	98.38 \pm 0.03(1.62)	100.00 \pm 0.00(0.00)	92.15 \pm 0.01(1.74)	100.00 \pm 0.00(0.00)	0.84	2.67 \pm 0.94
	CF-k	100.00 \pm 0.00(0.00)	0.06 \pm 0.03(99.94)	94.57 \pm 0.07(0.69)	38.43 \pm 1.05(61.57)	40.55	0.00 \pm 0.00
	SCRUB	93.15 \pm 9.26(6.85)	100.00 \pm 0.00(0.00)	86.16 \pm 7.88(7.17)	100.00 \pm 0.00(0.00)	3.51	18.67 \pm 7.41
	SSD	100.00 \pm 0.00(0.00)	65.27 \pm 1.43(34.73)	95.09 \pm 0.02(1.20)	100.00 \pm 0.00(0.00)	8.98	5.67 \pm 1.25
	RURK	99.80 \pm 0.04(0.20)	95.52 \pm 0.86(4.48)	93.93 \pm 0.06(0.04)	97.90 \pm 0.36(2.10)	1.70	1.00 \pm 0.00

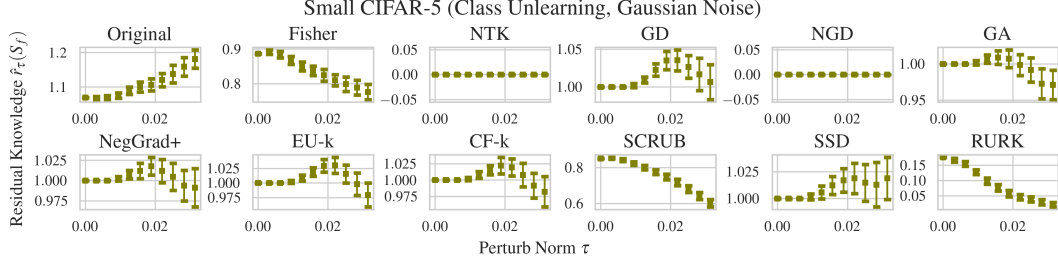


Figure C.8: Residual knowledge $\hat{r}_\tau(\mathcal{S}_f)$ of the proposed RURK, Original, and other unlearning methods on small CIFAR-5 with class unlearning, evaluated under varying perturbation norms τ , using Gaussian noise ($p = 2$) to draw $c = 100$ samples from $\mathcal{B}_p(\mathbf{x}, \tau)$.

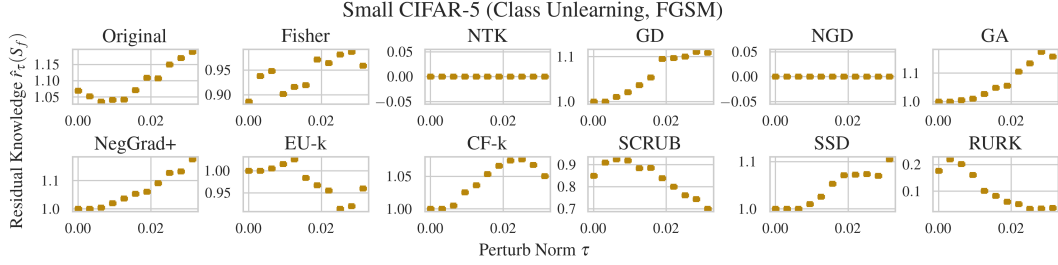


Figure C.9: Residual knowledge $\hat{r}_\tau(\mathcal{S}_f)$ of the proposed RURK, Original, and other unlearning methods on small CIFAR-5 with class unlearning, evaluated under varying perturbation norms τ , using targeted FGSM ($p = \infty$) to draw $c = 100$ samples from $\mathcal{B}_p(\mathbf{x}, \tau)$.

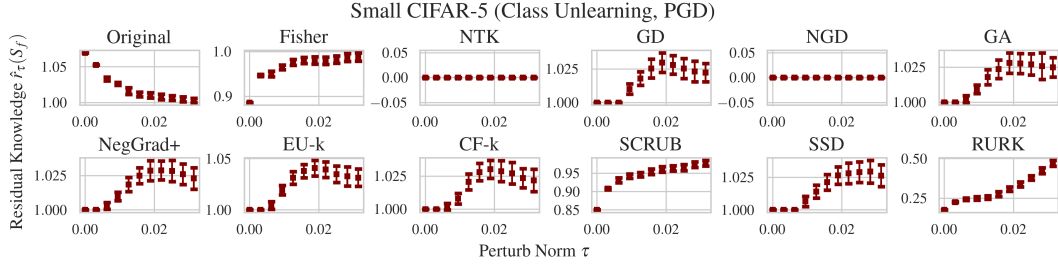


Figure C.10: Residual knowledge $\hat{r}_\tau(\mathcal{S}_f)$ of the proposed RURK, Original, and other unlearning methods on small CIFAR-5 with class unlearning, evaluated under varying perturbation norms τ , using targeted PGD ($p = \infty$) to draw $c = 100$ samples from $\mathcal{B}_p(\mathbf{x}, \tau)$.

975 C.3 Ablation studies on RURK

976 In this section, we conduct ablation studies on the key hyper-parameters of RURK, including the
 977 perturbation radius τ , regularization strength λ , sample size v , types of adversarial attacks, and
 978 alternative model architectures. We focus on the sample unlearning setting in the small CIFAR-5
 979 scenario, with all results summarized in Table C.4, Table C.5, Figure C.11, and Figure C.12.

980 **Different perturbation norms τ .** The perturbation norm τ determines the radius of the perturbation
 981 ball used to evaluate residual knowledge. We set $\tau = 0.03 \approx 8/255$, which aligns with common
 982 practice for the maximum adversarial perturbation. To assess the sensitivity of RURK to this parameter,
 983 we conduct an ablation study over $\tau \in \{0.00, 0.01, 0.02, 0.03, 0.04, 0.10\}$. As shown in Table C.4,
 984 when $\tau = 0.00$ or 0.01 , RURK fails to unlearn the forget samples from Original. However, as τ
 985 increases, the unlearning accuracy improves monotonically, while the test accuracy declines—as
 986 expected due to the increased regularization. Notably, RURK achieves the best average gap when
 987 $\tau = 0.03$. Since τ is the key hyper-parameter controlling the trade-off between unlearning efficacy
 988 and utility, we further visualize the estimated residual knowledge under varying τ in Figure C.11.
 989 The figure shows that increasing τ improves robustness against residual knowledge under stronger
 990 attacks. For instance, under Gaussian noise and FGSM, the residual knowledge remains near 1 for
 991 small τ . In contrast, PGD—a stronger adversary—can still uncover residual knowledge at low τ
 992 values, but its effectiveness diminishes significantly when $\tau = 0.1$, indicating improved robustness.

993 **Different regularization strengths λ .** The regularization strength λ controls the weighting of
 994 samples in the vulnerable set within the loss function defined in Eq. (6). We sweep over $\lambda \in$
 995 $\{0.00, 0.01, 0.02, 0.03, 0.04, 0.10\}$ and observe that the trend in the average gap closely mirrors that
 996 of τ . This similarity is expected, as both τ and λ jointly influence the extent to which residual
 997 knowledge is removed during unlearning.

998 **Other methods to search for the vulnerable set.** In the main text, we search for samples in the
 999 vulnerable set $\mathcal{V}((\mathbf{x}, y), \tau)$ using Gaussian noise. Here, we extend the evaluation to include stronger
 1000 adversarial methods, such as FGSM and PGD, with varying numbers of attack steps (indicated by the
 1001 number following PGD in Table C.4). We observe that FGSM—a targeted attack—achieves MIA
 1002 accuracy comparable to that of Re-train. Likewise, PGD with 5 steps also yields MIA accuracy
 1003 close to Re-train, indicating effective removal of forget-set information. These results suggest
 1004 that targeted attacks (FGSM, PGD) are more effective than untargeted ones like Gaussian noise in
 1005 eliminating residual knowledge, albeit at the cost of increased unlearning time.

1006 **Different samples size v .** The sample size v in Eq.(6) determines how extensively the vulnerable
 1007 set $\mathcal{V}((\mathbf{x}, y), \tau)$ is explored through sampling. In high-dimensional settings, $\mathcal{V}((\mathbf{x}, y), \tau)$ may contain
 1008 a large number of perturbed variants. However, due to computational constraints, it is impractical to
 1009 include all possible samples during training. Instead, we randomly draw v samples from $\mathcal{V}((\mathbf{x}, y), \tau)$
 1010 for each forget sample. We experiment with $v \in \{1, 2, 3, 4\}$ and observe that the average gap achieved
 1011 by RURK remains relatively stable across these values. This robustness is expected, as the loss term
 1012 $\kappa(\mathbf{w}, (\mathbf{x}, y))$ in Eq.(6) is computed as an average over the v sampled perturbations.

1013 **Other learning structures.** Thus far, our evaluation of RURK has focused on models trained with
 1014 ResNet-18. To assess its generality across architectures, we present results on small CIFAR-5 using
 1015 VGG-11 (Simonyan and Zisserman, 2014), as shown in Table C.5. Compared to NGD, RURK continues
 1016 to achieve the smallest average gap. Figure C.12 further illustrates the residual knowledge comparison
 1017 between RURK and NGD, demonstrating that RURK maintains lower and more stable residual knowledge
 1018 across perturbations. These results confirm that RURK effectively removes forget-set information and
 1019 controls residual knowledge across different network architectures, including both plain convolutional
 1020 networks (VGG-11) and those with residual connections (ResNet-18).

Table C.4: Ablation study of RURK on small CIFAR-5 with sample unlearning. Results are reported in the format $a \pm b$, indicating the mean a and standard deviation b over 3 independent trials. The absolute performance gap relative to Re-train is shown in (blue). For methods that fail to recover the forget-set knowledge within 30 training epochs, the re-learn time is reported as “>30”. We bold the results with hyper-parameters the same in the main text.

Other Parameters	Methods	Evaluation Metrics					Re-learn Time (# Epoch)
		Retain Acc. (%)	Unlearn Acc. (%)	Test Acc. (%)	MIA Acc. (%)	Avg. Gap	
-	Original	99.93 \pm 0.10(0.03)	0.00 \pm 0.00(8.33)	95.37 \pm 0.80(0.57)	4.67 \pm 3.30(22.33)	7.82	-
	Re-train	99.96 \pm 0.05(0.00)	8.33 \pm 3.30(0.00)	94.80 \pm 0.85(0.00)	27.00 \pm 5.66(0.00)	0.00	3.33 \pm 0.47
$\lambda = 0.03$ $v = 1$ # epoch = 1	RURK ($\tau = 0.00$)	99.93 \pm 0.10(0.03)	0.00 \pm 0.00(8.33)	95.10 \pm 0.99(0.30)	14.33 \pm 8.01(12.67)	5.33	1.00 \pm 0.00
	RURK ($\tau = 0.01$)	99.93 \pm 0.10(0.03)	0.00 \pm 0.00(8.33)	94.63 \pm 1.04(0.17)	15.67 \pm 8.96(11.33)	4.97	1.00 \pm 0.00
	RURK ($\tau = 0.02$)	99.89 \pm 0.16(0.07)	2.33 \pm 0.47(6.00)	94.27 \pm 1.08(0.53)	22.00 \pm 9.90(5.00)	2.90	1.00 \pm 0.00
	RURK ($\tau = 0.03$)	99.52\pm0.37(0.44)	5.67\pm2.36(2.66)	93.83\pm0.90(0.97)	33.33\pm12.26(6.33)	2.60	2.00\pm0.00
	RURK ($\tau = 0.04$)	98.96 \pm 0.37(1.00)	13.33 \pm 0.47(5.00)	92.37 \pm 1.23(2.43)	38.67 \pm 14.61(11.67)	5.03	1.67 \pm 0.47
	RURK ($\tau = 0.10$)	96.89 \pm 0.16(3.07)	29.67 \pm 4.71(21.34)	87.80 \pm 2.55(7.00)	54.00 \pm 15.56(27.00)	14.60	1.67 \pm 0.47
$\lambda = 0.03$ $\tau = 0.03$ $v = 1$ # epoch = 1	RURK ($\lambda = 0.00$)	99.96 \pm 0.05(0.00)	0.00 \pm 0.00(8.33)	95.00 \pm 1.13(0.20)	11.33 \pm 6.60(15.67)	6.05	1.00 \pm 0.00
	RURK ($\lambda = 0.01$)	99.93 \pm 0.10(0.03)	0.00 \pm 0.00(8.33)	94.60 \pm 0.99(0.20)	15.67 \pm 8.96(11.33)	4.97	1.00 \pm 0.00
	RURK ($\lambda = 0.02$)	99.89 \pm 0.16(0.07)	2.67 \pm 0.94(5.66)	94.17 \pm 1.08(0.63)	22.67 \pm 10.37(4.33)	2.68	1.00 \pm 0.00
	RURK ($\lambda = 0.03$)	99.52\pm0.37(0.44)	5.67\pm2.36(2.66)	93.83\pm0.90(0.97)	33.33\pm12.26(6.33)	2.60	2.00\pm0.00
	RURK ($\lambda = 0.04$)	98.89 \pm 0.47(1.07)	13.33 \pm 0.47(5.00)	92.50 \pm 0.85(2.30)	39.67 \pm 13.20(12.67)	5.26	1.67 \pm 0.47
	RURK ($\lambda = 0.10$)	85.04 \pm 2.62(14.92)	76.67 \pm 6.60(68.34)	76.87 \pm 0.05(17.93)	87.00 \pm 4.24(60.00)	40.30	2.00 \pm 0.00
$\lambda = 0.03$ $\tau = 0.03$ $v = 1$ # epoch = 1	RURK (Gaussian)	99.52\pm0.37(0.44)	5.67\pm2.36(2.66)	93.83\pm0.90(0.97)	33.33\pm12.26(6.33)	2.60	2.00\pm0.00
	RURK (FGSM)	99.70 \pm 0.10(0.26)	4.33 \pm 0.47(4.00)	94.17 \pm 1.08(0.63)	28.67 \pm 11.79(1.67)	1.64	1.00 \pm 0.00
	RURK (PGD 1)	98.81 \pm 0.10(1.15)	12.33 \pm 0.47(4.00)	92.53 \pm 1.32(2.27)	46.67 \pm 14.61(19.67)	6.77	1.67 \pm 0.47
	RURK (PGD 5)	99.89 \pm 0.05(0.11)	2.00 \pm 0.00(6.33)	94.07 \pm 1.23(0.73)	27.33 \pm 13.67(0.33)	1.88	1.00 \pm 0.00
	RURK (PGD 10)	99.93 \pm 0.10(0.03)	0.67 \pm 0.47(7.66)	94.57 \pm 1.08(0.23)	20.00 \pm 11.31(7.00)	3.73	1.00 \pm 0.00
	Gaussian	99.52\pm0.37(0.44)	5.67\pm2.36(2.66)	93.83\pm0.90(0.97)	33.33\pm12.26(6.33)	2.60	2.00\pm0.00
$\lambda = 0.03$ # epoch = 1	RURK ($v = 2$)	99.52 \pm 0.37(0.44)	4.67 \pm 0.94(3.66)	93.87 \pm 0.94(0.93)	33.33 \pm 12.26(6.33)	2.84	1.33 \pm 0.47
	RURK ($v = 3$)	99.52 \pm 0.37(0.44)	5.00 \pm 1.41(3.33)	93.87 \pm 0.94(0.93)	33.33 \pm 12.26(6.33)	2.76	1.33 \pm 0.47
	RURK ($v = 4$)	99.52 \pm 0.37(0.44)	5.67 \pm 2.36(2.66)	93.83 \pm 0.90(0.97)	33.33 \pm 12.26(6.33)	2.60	1.00 \pm 0.00

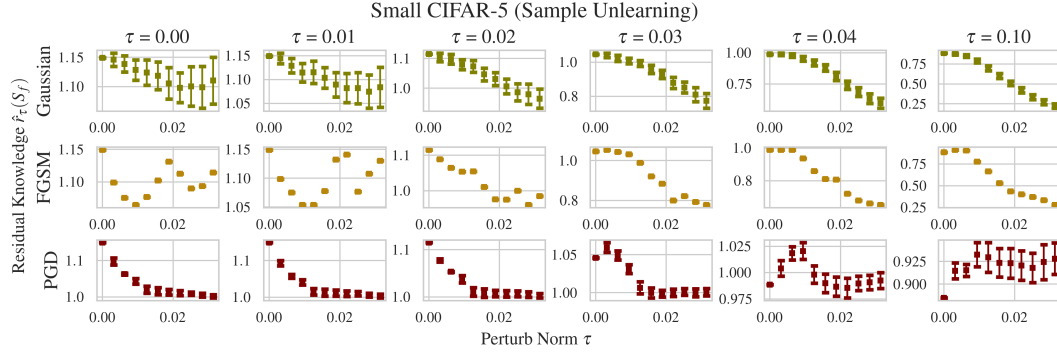


Figure C.11: Residual knowledge $\hat{r}_\tau(\mathcal{S}_f)$ of the proposed RURK, Original, and other unlearning methods on small CIFAR-5 with sample unlearning, evaluated under varying perturbation norms τ , using untargeted Gaussian noise ($p = 2$), targeted FGSM ($p = \infty$), and targeted PGD ($p = \infty$) to draw $c = 100$ samples from $\mathcal{B}_p(\mathbf{x}, \tau)$.

Table C.5: The performance summary of various unlearning methods on small CIFAR-5 with VGG-11. Results are reported in the format $a \pm b$, indicating the mean a and standard deviation b over 3 independent trials. The absolute performance gap relative to Re-train is shown in (blue). For methods that fail to recover the forget-set knowledge within 30 training epochs, the re-learn time is reported as “>30”.

Datasets	Methods	Evaluation Metrics					
		Retain Acc. (%)	Unlearn Acc. (%)	Test Acc. (%)	MIA Acc. (%)	Avg. Gap	Re-learn Time (# Epoch)
Small CIFAR-5	Original	100.00 \pm 0.00	0.00 \pm 0.00	94.10 \pm 0.00	0.00 \pm 0.00	9.01	-
	Re-train	97.44 \pm 0.00	15.00 \pm 0.00	91.60 \pm 0.00	16.00 \pm 0.00	0.00	4.33 \pm 0.47
	NGD	95.00 \pm 0.00	3.00 \pm 0.00	91.40 \pm 0.00	1.00 \pm 0.00	7.41	1.00 \pm 0.00
	RURK	99.33 \pm 0.00	7.00 \pm 0.00	92.70 \pm 0.00	22.00 \pm 0.00	4.25	2.00 \pm 0.00

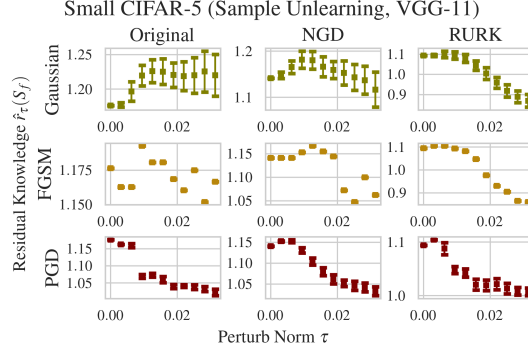


Figure C.12: Residual knowledge $\hat{r}_\tau(S_f)$ of the proposed RURK, Original, and NGD on small CIFAR-5 with sample unlearning and VGG-11 structure, evaluated under varying perturbation norms τ , using untargeted Gaussian noise ($p = 2$), targeted FGSM ($p = \infty$), and targeted PGD ($p = \infty$) to draw $c = 100$ samples from $\mathcal{B}_p(\mathbf{x}, \tau)$.

C.4 The prevalence of residual knowledge

In Figure 2, we report the estimated residual knowledge over the entire forget set $\hat{r}_\tau(S_f)$. We also provide per-sample estimates $\hat{r}_\tau((\mathbf{x}, y))$ for each $(\mathbf{x}, y) \in S_f$, as defined in Eq. (4). Table C.6 summarizes the proportion of forget samples exhibiting residual knowledge greater than one—i.e., those still recognizable after unlearning—under different perturbation norms τ , evaluated on small CIFAR-5 (with NTK) and CIFAR-10 (with NGD). Remarkably, even under imperceptibly small perturbations (e.g., $\tau = 0.013$), about 11% of the forget samples in small CIFAR-5 and more than 8% (approximately 160 samples) in CIFAR-10 still exhibit residual knowledge above one. These findings highlight that residual knowledge is not only prevalent but also poses a significant privacy risk, emphasizing the need for stronger certification and mitigation techniques in machine unlearning.

Table C.6: Percentage of forget samples that have residual knowledge $\hat{r}_\tau((\mathbf{x}, y))$ large than 1.

Datasets	Methods	Gaussian Perturbation Norm τ										
		0.000	0.003	0.006	0.009	0.013	0.016	0.019	0.022	0.025	0.028	0.031
Small CIFAR-5	NTK	0.00 \pm 0.00	3.00 \pm 1.71	5.00 \pm 2.18	6.00 \pm 2.37	11.00 \pm 3.13	14.00 \pm 3.47	26.00 \pm 4.39	34.00 \pm 4.74	39.00 \pm 4.88	45.00 \pm 4.97	48.00 \pm 5.00
CIFAR-10	NGD	0.00 \pm 0.00	1.85 \pm 1.35	5.25 \pm 2.23	7.25 \pm 2.59	8.50 \pm 2.79	8.55 \pm 2.80	9.30 \pm 2.90	8.90 \pm 2.85	8.10 \pm 2.73	7.10 \pm 2.57	6.95 \pm 2.54

1031 NeurIPS Paper Checklist

1032 1. Claims

1033 Question: Do the main claims made in the abstract and introduction accurately reflect the
1034 paper's contributions and scope?

1035 Answer: [\[Yes\]](#)

1036 Justification: We include a summary of contributions at the end of the introduction, where
1037 each contribution and the claims made therein are specifically referred to a section in this
1038 paper.

1039 Guidelines:

- 1040 • The answer NA means that the abstract and introduction do not include the claims
1041 made in the paper.
- 1042 • The abstract and/or introduction should clearly state the claims made, including the
1043 contributions made in the paper and important assumptions and limitations. A No or
1044 NA answer to this question will not be perceived well by the reviewers.
- 1045 • The claims made should match theoretical and experimental results, and reflect how
1046 much the results can be expected to generalize to other settings.
- 1047 • It is fine to include aspirational goals as motivation as long as it is clear that these goals
1048 are not attained by the paper.

1049 2. Limitations

1050 Question: Does the paper discuss the limitations of the work performed by the authors?

1051 Answer: [\[Yes\]](#)

1052 Justification: We include a discussion of limitations regarding theoretical extensions and
1053 computational overhead in the last section.

1054 Guidelines:

- 1055 • The answer NA means that the paper has no limitation while the answer No means that
1056 the paper has limitations, but those are not discussed in the paper.
- 1057 • The authors are encouraged to create a separate "Limitations" section in their paper.
- 1058 • The paper should point out any strong assumptions and how robust the results are to
1059 violations of these assumptions (e.g., independence assumptions, noiseless settings,
1060 model well-specification, asymptotic approximations only holding locally). The authors
1061 should reflect on how these assumptions might be violated in practice and what the
1062 implications would be.
- 1063 • The authors should reflect on the scope of the claims made, e.g., if the approach was
1064 only tested on a few datasets or with a few runs. In general, empirical results often
1065 depend on implicit assumptions, which should be articulated.
- 1066 • The authors should reflect on the factors that influence the performance of the approach.
1067 For example, a facial recognition algorithm may perform poorly when image resolution
1068 is low or images are taken in low lighting. Or a speech-to-text system might not be
1069 used reliably to provide closed captions for online lectures because it fails to handle
1070 technical jargon.
- 1071 • The authors should discuss the computational efficiency of the proposed algorithms
1072 and how they scale with dataset size.
- 1073 • If applicable, the authors should discuss possible limitations of their approach to
1074 address problems of privacy and fairness.
- 1075 • While the authors might fear that complete honesty about limitations might be used by
1076 reviewers as grounds for rejection, a worse outcome might be that reviewers discover
1077 limitations that aren't acknowledged in the paper. The authors should use their best
1078 judgment and recognize that individual actions in favor of transparency play an impor-
1079 tant role in developing norms that preserve the integrity of the community. Reviewers
1080 will be specifically instructed to not penalize honesty concerning limitations.

1081 3. Theory Assumptions and Proofs

1082 Question: For each theoretical result, does the paper provide the full set of assumptions and
1083 a complete (and correct) proof?

Answer: [Yes]

Justification: We clearly state all the assumptions and lemmas that we have used with citations in our theoretical results. We also provide the sketch of proof/intuition in the main text.

Guidelines:

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

4. Experimental Result Reproducibility

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

Answer: [Yes]

Justification: We provide details descriptions of the proposed methodology in the main text and in the appendix. Our methodology is tested on over 2 different vision benchmarks, shows consistency and explainable results over independent trials, and outperforms 11 existing unlearning baseline algorithms.

Guidelines:

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

1138 some way (e.g., to registered users), but it should be possible for other researchers
1139 to have some path to reproducing or verifying the results.

1140 5. Open access to data and code

1141 Question: Does the paper provide open access to the data and code, with sufficient instruc-
1142 tions to faithfully reproduce the main experimental results, as described in supplemental
1143 material?

1144 Answer: [No]

1145 Justification: Due to intellectual property protection and anonymity requirements, we choose
1146 to release our codes upon decision. We provide details scripts on how to access the datasets,
1147 implement our methodology, and reproduce the empirical results in the main text and
1148 appendix.

1149 Guidelines:

- 1150 • The answer NA means that paper does not include experiments requiring code.
- 1151 • Please see the NeurIPS code and data submission guidelines ([https://nips.cc/
1152 public/guides/CodeSubmissionPolicy](https://nips.cc/public/guides/CodeSubmissionPolicy)) for more details.
- 1153 • While we encourage the release of code and data, we understand that this might not be
1154 possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not
1155 including code, unless this is central to the contribution (e.g., for a new open-source
1156 benchmark).
- 1157 • The instructions should contain the exact command and environment needed to run to
1158 reproduce the results. See the NeurIPS code and data submission guidelines ([https:
1159 //nips.cc/public/guides/CodeSubmissionPolicy](https://nips.cc/public/guides/CodeSubmissionPolicy)) for more details.
- 1160 • The authors should provide instructions on data access and preparation, including how
1161 to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- 1162 • The authors should provide scripts to reproduce all experimental results for the new
1163 proposed method and baselines. If only a subset of experiments are reproducible, they
1164 should state which ones are omitted from the script and why.
- 1165 • At submission time, to preserve anonymity, the authors should release anonymized
1166 versions (if applicable).
- 1167 • Providing as much information as possible in supplemental material (appended to the
1168 paper) is recommended, but including URLs to data and code is permitted.

1169 6. Experimental Setting/Details

1170 Question: Does the paper specify all the training and test details (e.g., data splits, hyper-
1171 parameters, how they were chosen, type of optimizer, etc.) necessary to understand the
1172 results?

1173 Answer: [Yes]

1174 Justification: We provide all the training and setting details in the main text and appendix,
1175 including hyper-parameter settings, optimizer/learning rate scheduler, the GitHub links to
1176 all unlearning baselines we have evaluated, the reference to the evaluation metrics, etc.

1177 Guidelines:

- 1178 • The answer NA means that the paper does not include experiments.
- 1179 • The experimental setting should be presented in the core of the paper to a level of detail
1180 that is necessary to appreciate the results and make sense of them.
- 1181 • The full details can be provided either with the code, in appendix, or as supplemental
1182 material.

1183 7. Experiment Statistical Significance

1184 Question: Does the paper report error bars suitably and correctly defined or other appropriate
1185 information about the statistical significance of the experiments?

1186 Answer: [Yes]

1187 Justification: All the results reported in this paper include error bars that is computed over
1188 3 independent trials. Moreover, we implement 5 different evaluation metrics (operational
1189 meaning and proper reference included) to compare our method against existing baselines.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments Compute Resources

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

Answer: [Yes]

Justification: We provide the specific type of machine we used for the experiments in the appendix. We also include a discussion of the computational complexity of our method in § 5 and in the appendix.

Guidelines:

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

9. Code Of Ethics

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

Answer: [Yes]

Justification: We have reviewed and complied the NeurIPS Code of Ethics.

Guidelines:

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

10. Broader Impacts

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

1242 Answer: [Yes]

1243 Justification: We cover the societal impacts of the residual knowledge and machine unlearn-

1244 ing in the introduction, related work and final remark.

1245 Guidelines:

- 1246 • The answer NA means that there is no societal impact of the work performed.
- 1247 • If the authors answer NA or No, they should explain why their work has no societal
- 1248 impact or why the paper does not address societal impact.
- 1249 • Examples of negative societal impacts include potential malicious or unintended uses
- 1250 (e.g., disinformation, generating fake profiles, surveillance), fairness considerations
- 1251 (e.g., deployment of technologies that could make decisions that unfairly impact specific
- 1252 groups), privacy considerations, and security considerations.
- 1253 • The conference expects that many papers will be foundational research and not tied
- 1254 to particular applications, let alone deployments. However, if there is a direct path to
- 1255 any negative applications, the authors should point it out. For example, it is legitimate
- 1256 to point out that an improvement in the quality of generative models could be used to
- 1257 generate deepfakes for disinformation. On the other hand, it is not needed to point out
- 1258 that a generic algorithm for optimizing neural networks could enable people to train
- 1259 models that generate Deepfakes faster.
- 1260 • The authors should consider possible harms that could arise when the technology is
- 1261 being used as intended and functioning correctly, harms that could arise when the
- 1262 technology is being used as intended but gives incorrect results, and harms following
- 1263 from (intentional or unintentional) misuse of the technology.
- 1264 • If there are negative societal impacts, the authors could also discuss possible mitigation
- 1265 strategies (e.g., gated release of models, providing defenses in addition to attacks,
- 1266 mechanisms for monitoring misuse, mechanisms to monitor how a system learns from
- 1267 feedback over time, improving the efficiency and accessibility of ML).

1268 **11. Safeguards**

1269 Question: Does the paper describe safeguards that have been put in place for responsible

1270 release of data or models that have a high risk for misuse (e.g., pretrained language models,

1271 image generators, or scraped datasets)?

1272 Answer: [NA]

1273 Justification: [NA]

1274 Guidelines:

- 1275 • The answer NA means that the paper poses no such risks.
- 1276 • Released models that have a high risk for misuse or dual-use should be released with
- 1277 necessary safeguards to allow for controlled use of the model, for example by requiring
- 1278 that users adhere to usage guidelines or restrictions to access the model or implementing
- 1279 safety filters.
- 1280 • Datasets that have been scraped from the Internet could pose safety risks. The authors
- 1281 should describe how they avoided releasing unsafe images.
- 1282 • We recognize that providing effective safeguards is challenging, and many papers do
- 1283 not require this, but we encourage authors to take this into account and make a best
- 1284 faith effort.

1285 **12. Licenses for existing assets**

1286 Question: Are the creators or original owners of assets (e.g., code, data, models), used in

1287 the paper, properly credited and are the license and terms of use explicitly mentioned and

1288 properly respected?

1289 Answer: [Yes]

1290 Justification: We provide proper citations for all datasets including licenses and how to

1291 access, and programming packages (Python, Pytorch, Scikit Learn, torchattack) used in this

1292 paper.

1293 Guidelines:

- 1294 • The answer NA means that the paper does not use existing assets.

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

13. New Assets

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

Answer: [NA]

Justification: [NA]

Guidelines:

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

14. Crowdsourcing and Research with Human Subjects

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

Answer: [NA]

Justification: [NA]

Guidelines:

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

15. Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects

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

Answer: [NA]

Justification: [NA]

1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356

Guidelines:

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