New Frontiers for Machine Unlearning

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
in
Computer Science and Engineering by Research

by

Shashwat Goel
2019111006
shashwat.goel@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
March 2024

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled **"New Frontiers for Machine Unlearning"** by Shashwat Goel, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Adviser: Prof. Ponnurangam Kumaraguru

To Mom and Dad

Acknowledgments

Embarking on this journey, I have been fortunate enough to be surrounded by individuals whose support, guidance, and encouragement have been pivotal to the completion of this thesis. It is with a heart full of gratitude that I take this moment to acknowledge their invaluable contributions.

First and foremost, I thank my advisor Dr. Ponnurangam Kumaraguru. His unwavering commitment to keep me accountable, coupled with the flexibility in allowing me to explore my interests and collaborate with scholars outside our institution, has been foundational to my research journey. His patience and belief in my abilities have helped me get through rough patches. I have learnt a lot both personally and professionally from just watching PK masterfully managing our research group.

I thank my parents, whose relentless encouragement and support have been my stronghold. They have consistently pushed me to excel, ensuring that I had everything needed to focus solely on my research. Their sacrifices have not gone unnoticed, and I am grateful for their love.

I am immensely grateful to some seniors whose mentorship has helped me grow as a researcher, and whose advice has been invaluable. First, Ameya Prabhu, whose mentorship took me from a novice in Deep Learning to wherever I stand today. His guidance in picking interesting problems that are well suited for junior researchers to learn quickly was fundamental in ensuring a good start to my research journey. Ameya has constantly pushed me to write better code, and his research has been an inspiration. Conversations with him have been invaluable in shaping my research style. I thank Dr. Amartya Sanyal, who has taught me to reason critically about everything I write, which has often led to reconsiderations and updates in my beliefs. This collaboration could not have happened without Shyam Gopal Karthik making the initial connection with Ameya, guiding me through the most important process of picking my first problem statement and advisor. I thank Nathaniel Li, for being an extremely supportive collaborator, and helping ensure I was comfortable throughout my time working with the Center for AI Safety. I thank Dr. Dan Hendrycks, whose discussions on doing impactful research have changed my perspective completely in so many ways. Finally, I thank Saujas Vaduguru, who has been an ideal mentor, patiently listening to my rants during hard times, and keeping me grounded. It was a pleasure to be able to collaborate with Saujas on the negation project, and if my academic writing has become any better over the years, it is thanks to Saujas.

My gratitude also extends to my friends and collaborators, Nikhil Chandak and Shashwat Singh. Your willingness to serve as a sounding board for my ideas, providing insightful feedback, continuing to drive progress and push me when I slacked off, has made my research experiences far more enjoyable. I thank

my friends Varshita, Kartik, and Meghna, for making college life not just bearable but also memorable. Your support has made all the difference, smoothing out the rough patches and making the good times even better.

Finally, I thank you, the reader. I hope you find this work interesting.

Abstract

Machine Learning models increasingly face data integrity challenges due to the use of large-scale training datasets drawn from the Internet. We study what model developers can do if they detect that some data was manipulated or incorrect. Such manipulated data can cause adverse effects like vulnerability to backdoored samples, systematic biases, and in general, reduced accuracy on certain input domains. Machine Unlearning, traditionally studied for handling user data-deletion requests to provide privacy, can address these by allowing post-hoc deletion of affected training data from a learned model. Achieving perfect unlearning is computationally expensive; consequently, prior works have proposed inexact unlearning algorithms to solve this approximately as well as evaluation methods to test the effectiveness of these algorithms.

In this thesis, we first outline some necessary criteria for evaluation methods and show no existing evaluation satisfies them all. Then, we design a stronger black-box evaluation method called the Interclass Confusion (IC) test which adversarially manipulates data during training to detect the insufficiency of unlearning procedures. We also propose two analytically motivated baseline methods (EU-k and CF-k) which outperform several popular inexact unlearning methods. We demonstrate how adversarial evaluation strategies can help in analyzing various unlearning phenomena which can guide the development of stronger unlearning algorithms.

Next, we study the practical constraint that model developers may not know all manipulated training samples. Often, only a small, representative subset of the affected data is flagged. We formalize "Corrective Machine Unlearning" as the problem of mitigating the impact of data affected by unknown manipulations on a trained model, possibly knowing only a subset of impacted samples. We demonstrate that the problem of corrective unlearning has significantly different requirements from traditional privacy-oriented unlearning. We find most existing unlearning methods, including the gold-standard retraining-from-scratch, require most of the manipulated data to be identified for effective corrective unlearning. However, one approach, SSD, achieves limited success in unlearning adverse effects with just a small portion of the manipulated samples, showing the tractability of this setting. We hope our work spurs research towards developing better methods for corrective unlearning.

Finally, we demonstrate the use of unlearning in reducing the risk of Large Language Models assisting malicious use in the creation of bioweapons and cyberattacks. Adaptations of existing state-of-the-art unlearning techniques fail on this task, probably due to complexities introduced by not having access to training data that leads to such capabilities. We discuss Contrastive Unlearning Tuning (CUT), a

Representation Engineering based unlearning method that steers models towards novice behaviour on potentially harmful dual-use knowledge, while retaining general model capabilities. We design a probing evaluation which shows CUT succeeds in removing this knowledge even from the internal layer representations of LLMs.

Overall, this thesis attempts to extend the frontiers of unlearning from user-privacy applications to debiasing, denoising, removing backdoors, and removing harmful dual-use capabilities. We highlight the shortcomings of privacy-oriented unlearning methods and formulations in achieving these goals. We hope our work offers practitioners a new strategy to handle challenges arising from web-scale training, and post-training line of defense towards ensuring AI Safety.

Contents

List of Figures

List of Tables

Chapter *1*

Introduction

Deep learning has become increasingly prevalent in everyday applications, with models being trained on large amounts of sensitive personal information including health and financial records, social network history, personal emails, and messages. This has led to growing privacy concerns, as codified in privacy legislation like GDPR (Council of European Union, 2018), CCPA (California State Leglisature, 2018), and PIPEDA (Parliament of Canada, 2018). The underlying motivation for privacy legislation is the concept of *data autonomy*, which states that every individual must retain complete control of their own data, including the right to withdraw their data from any system. However, deleting records corresponding to individuals is considerably harder for machine learning systems, especially those using deep networks, than with traditional databases.

Studies, such as (Feldman and Zhang, 2020; Zhang et al., 2017), have shown that deep neural networks have a tendency to *memorize* data. This means that the network not only learns common patterns in the data, but also stores information about individual training data points. This is concerning from a privacy standpoint, as this information can be detected (Shokri et al., 2017) or even extracted from the model (Carlini et al., 2019). The main goal of "machine unlearning" (Federal Trade Commission, 2021; Cao and Yang, 2015; Ginart et al., 2019; Bourtoule et al., 2021; VIL, 2018) is to design both algorithms to delete data stored in the network and evaluation methods to recover or detect the deleted data from the trained model. Preserving privacy and removing memorization are not the only motivations to study machine unlearning. In this thesis, we focus on extending the frontiers of unlearning by proposing its use for the removal of incorrect data, manipulated data, and dual-use knowledge from ML models.

With the increasing prevalence of models being trained on web-scale datasets, collected with loose quality controls, we argue removing the effects of incorrect or manipulated data is an important avenue for Machine Unlearning. Several studies have shown that small amounts of corrupted data can induce harmful properties into the trained model, which can greatly affect its behaviour on unseen data Nakkiran and Bansal (2020), a phenomenon we refer to as *property generalization*. This can lead to problems in trustworthy machine learning, such as with noisy data Frenay and Verleysen (2014); Northcutt et al.

(2021b,a), systematically biased data Prabhu and Birhane (2021)[1], or adversarial data, such as poisoned samples Barreno et al. (2006); Chen et al. (2017); Yang et al. (2020). For example, Sanyal et al. (2021); Paleka and Sanyal (2023) show that a small amount of random noisy labels can significantly harm the adversarial robustness of a model. Further, Konstantinov and Lampert (2022) show that a small set of adversarially corrupted data can greatly increase unfairness. As these corrupted samples are discovered, model developers can use unlearning to remove the unwanted properties induced in previously trained models.

Large Language Models (LLMs) trained on web-scale datasets often learn harmful dual-use knowledge. The White House Executive Order on Artificial Intelligence (White House, 2023) mentions concerns of AI being misused to develop chemical, biological, radiological, nuclear, and cyber weapons. For instance, AI coding assistants can make it easier for non-experts to execute cyberattacks (Fang et al., 2024) with increased stealth and scale, which if targeted at infrastructure like power grids can lead to massive harm (UK Cabinet Office, 2023). Similar risks exist for reducing barriers to biological weapon development (Sandbrink, 2023). Unlearning potentially dual-use knowledge from widely accessible versions of AI models can act as one mitigation strategy towards such risks. Unlearned models have higher inherent safety: if the model lacks the hazardous knowledge necessary to enable malicious use, it will be safe even if jailbroken (Zou et al., 2023b).

Roadmap: In Chapter 2, we first argue for the need of stronger unlearning evaluations. Particularly, we show shortcomings of existing evaluations, and propose the Interclass Confusion test as an adversarial evaluation that demonstrates the weakness of some existing unlearning procedures. In Chapter 3, we then discuss the problem of corrective unlearning, that is removing the influence of wrong or manipulated training data, and how it has different requirements from unlearning for privacy, requiring separate treatment. We show that state of the art unlearning methods are insufficient for corrective unlearning, as often all the manipulated data is not known. Finally, in Chapter 4 we show how unlearning can be used to remove potentially harmful dual-use knowledge and capabilities from Large Language Models.

[1] to the extent that bias is a dataset problem Hooker (2021).

* equal contribution

Chapter *2*

Adversarial Evaluations for Inexact Unlearning[1]

2.1 Introduction

The goal of Machine Unlearning is traditionally formalized using the concept of *model indistinguishability*, first defined in Golatkar et al. (2020a). Let M be an ML model trained on dataset S using learning algorithm T and $S_f \subset S$ be the set of points that need to be deleted from M. An unlearning process is considered successful if the distribution[2] of models produced by the unlearning process, is indistinguishable from the distribution of models produced by retraining the model using any training process T' on the remaining data $S \setminus S_f$. To see why model indistinguishability implies unlearning, note that no training procedure T' which only uses $S \setminus S_f$ can produce a model that carries information specific to S_f. Hence, it is sufficient to show the model distribution produced by the unlearning algorithm is indistinguishable from the model distribution produced by any one training algorithm T' using $S \setminus S_f$. In this work, we study deleting a single query batch of samples. Extension to sequential deletion has to tackle challenges like correlated queries across time (Chourasia et al., 2023).

A naive method for unlearning data from a machine learning model is to retrain the model on the retain data $S \setminus S_f$. This method removes all information from the deletion set. Hence, in theory, it achieves "exact unlearning", but is computationally and memory intensive. Our work focuses on "inexact unlearning", in which the goal is to unlearn most information from the deleted data while minimizing computational cost. While exact unlearning is often infeasible, inexact unlearning presents a more tractable objective. In the specific case of deep networks, due to the absence of theoretical guarantees, empirical tests are commonly used for evaluating the degree of unlearning. A strong empirical test should reliably distinguish models unlearning to varying degrees in terms of memorization and property generalization of the deletion set. The latter is challenging with existing evaluations which remove Independently Identically Distributed (I.I.D) samples as the undesirable properties they induce may be apriori unknown.

[1] Goel, S., Prabhu, A., Sanyal, A., Lim, S. N., Torr, P., Kumaraguru, P. (2022). Towards adversarial evaluations for inexact machine unlearning. arXiv preprint arXiv:2201.06640. All figures in this section are taken from the paper.

[2] due to stochasticity in both the unlearning algorithm and T

As comparing a distribution of ML models is intractable, most past evaluations compare the weights (Wu et al., 2020; Izzo et al., 2021) or outputs (Golatkar et al., 2020a,b, 2021; Peste et al., 2021) of an unlearnt model and one retrained using the original training procedure on $S \setminus S_f$. However, as we argue in Theorem 1, even achieving nearly identical weights is insufficient to guarantee similarity in even well known properties like adversarial error and fairness, and thus model indistinguishability. This motivates the need for adversarial evaluations of unlearning. We propose performing manipulation in the training data which introduces a known measurable property through S_f that is absent in $S \setminus S_f$. Thus, models that exhibit this property cannot be indistinguishable from models obtained through retraining on $S \setminus S_f$. Ideally, the property unique to S_f should produce a large predictable change in model behaviour to make its presence easy to measure. To this end, inspired by the application of removing systematically biased data, we propose the Interclass Confusion (IC) test. It induces the property of confusion between two classes through label manipulations. IC test requires the unlearning procedure to erase the induced confusion which we measure as the number of samples of the two classes "confused" as belonging to the other class. As discussed in the following sections, we can use this test to detect both memorisation and property generalisation in an efficient way.

We find that our proposed IC test is far stronger than existing evaluations, allowing us to glean interesting insights into unlearning algorithms. Using the IC test, we can demonstrate the insufficiency of a class of unlearning methods that simply modifies the final linear layer (Izzo et al., 2021; Baumhauer et al., 2022) in deep networks or methods that do not use the retained data $S \setminus S_f$ (Chundawat et al., 2023b). Our test detects the presence of imperfectly unlearnt information about the deletion set S_f in the early layers of a deep network. Along with designing a stronger evaluation method (IC test), we also present two strong novel baselines — EU-k, which retrains the last k layers from scratch and CF-k, where a model's last k layers are continually trained on the retain set $S \setminus S_f$. Finally, we also propose strategies to make the original model M more amenable to unlearning, thereby aiding faster unlearning.

Overall, in this chapter, we emphasise empirical evaluations of inexact-unlearning which measure how well an unlearning procedure forgets additional properties induced by the deletion set S_f. The proposed IC test alleviates certain shortcomings in existing evaluations as passing the IC test is necessary for achieving model indistinguishability. Further, it's adversarial nature makes it a much stronger test to pass than prior evaluations as shown by our experiments. The main contributions of this chapter are:

1. In line with the motivations of machine unlearning, we decompose the evaluation of unlearning into memorization and property generalisation. The former is computed on the forget set S_f whereas the latter is computed using unseen samples from the test set.
2. We highlight some necessary principles for useful evaluations of unlearning not achieved in existing work. We alleviate this by introducing a new black-box evaluation called the Interclass Confusion (IC) test. We empirically demonstrate that the IC test is far stronger than existing tests.
3. Further, we use the IC test to show several surprising phenomena which may guide the design of future unlearning methods (i) Unlearning just the last layer only removes a small fraction of

information about S_f (ii) Unlearning methods may require the ability to learn *i.e.*gain information (iii) Standard regularisation during training can make models more amenable to unlearning. Using these insights, we propose two strong baselines, EU-k and CF-k, for comparing future unlearning methods.

Roadmap: The rest of the chapter is organized as follows: Section 2.2 describes our proposed evaluation methods in context of prior work, Section 2.3 describes our unlearning baselines and their properties, Section 2.4 presents our experimental results.

2.2 Towards Adversarial Evaluations

We begin with an analysis of the shortcomings of prior evaluation strategies, and alleviate them by proposing the Interclass Confusion Test.

2.2.1 Trends in Unlearning Evaluations

In this section, we look at existing approaches for evaluating unlearning procedures. First, these methods choose one of the two types of deletion sets S_f: n I.I.D samples (I.I.D removal) (Golatkar et al., 2021; Wu et al., 2020; Izzo et al., 2021; Peste et al., 2021; He et al., 2021; Shibata et al., 2021) or n samples belonging to a particular class (Class Removal) (Golatkar et al., 2020a; Baumhauer et al., 2022). Once the unlearning procedure is applied on the above S_f, the following are some popular metrics used to measure forgetting:

Relearn Time: Golatkar et al. (2020a,b, 2021); Chundawat et al. (2023b) measure the number of training epochs until the loss of an unlearnt model drops below a pre-chosen threshold when retrained on samples in S_f. A higher re-learn time implies better forgetting.

Weight Similarity: Wu et al. (2020); Izzo et al. (2021) measure the L_2 distance between the weights of the unlearnt model and another model retrained on $S \setminus S_f$ using the original training procedure. Naturally, a smaller distance is used to imply better unlearning.

Output similarity: Similar to distance between weights, distance in the softmax outputs on a pre-defined set of data points are also used by different evaluation methods. Golatkar et al. (2020a,b, 2021) measures the L_1 distances between softmax outputs of a unlearnt model and a retrained model on the pre-defined set, Peste et al. (2021) measure the L_1 distance between the confusion matrices of the respective models, He et al. (2021); Golatkar et al. (2021); Shibata et al. (2021) measure the gap in error on the distribution of affected samples.

Membership Inference Attacks (MIA): Tests based on Membership Inference Attacks (Shokri et al., 2017; Song and Mittal, 2021) are designed to reliably distinguish data points in the training set from similar unseen data. Hence, they can also be used to reliably distinguish the deleted samples from similar unseen samples (see Hu et al. (2021) for a survey). A detailed description of our MIA attack compared to past unlearning literature is included in Section 2.5.2.

Table 2.1: Comparison of evaluation methods (sampling strategy+metric) in inexact unlearning. Only our IC test satisfies all three desiderata.

Deletion Set Sampling Strategy	Metric	Necessary for Indistinguishability	Comparable Across Training Procedures	Checks Property Generalization
I.I.D (Golatkar et al., 2021), Class (Golatkar et al., 2020a,b, 2021)	Relearn time	✓	×	×
I.I.D (Wu et al., 2020; Izzo et al., 2021; Thudi et al., 2021)	L2 Weights	×	×	✓
I.I.D (Peste et al., 2021)	L1-ConfusionMatrix	×	✓	✓
I.I.D (Golatkar et al., 2021), Class (Golatkar et al., 2020b, 2021)	L1-Softmax	×	✓	✓
Class (Golatkar et al., 2021, 2020b; Baumhauer et al., 2022), I.I.D (Golatkar et al., 2021; Ma et al., 2023)	MIA	✓	✓	×
I.I.D (He et al., 2021; Golatkar et al., 2021; Shibata et al., 2021), Class (Golatkar et al., 2020a,b, 2021)	Error	×	✓	✓
Interclass Confusion (Ours)	MIA	✓	✓	×
Interclass Confusion (Ours), I.I.D Confusion (Ours: Ablation)	Error	✓	✓	✓

2.2.2 Shortcomings of Existing Evaluations

We start by listing three desiderata absent in most existing evaluation methods, as summarized in Table 2.1. Theorem 1 then motivates the need for adversarial evaluations.

Necessary for Indistinguishability: Suppose there exists $T' \neq T$ such that retraining with T' on $S \setminus S_f$ would produce a model highly similar to the unlearnt model M_u, then M_u is a correct solution as it satisfies model indistinguishability. Showing no such T' exists is difficult, and thus past evaluations simply compare M_u with a model M_r^T retrained using the original training procedure. However, this can exclude a large set of correct solutions which have no information from S_f but behave differently from M_r^T. Consider a randomly initialized network. It clearly has no information from S_f, and indeed satisfies model indistinguishability if we consider T' to be the random initialization process. However, it will be arbitrarily far from M_r^T and will be unnecessarily penalized by evaluations based on Weights and Output similarity. Thus, passing past evaluations based on high similarity with a single model is not necessary for achieving unlearning.

On the other hand, our proposed evaluation manipulates a subset of training data to introduce a measurable property through S_f that is absent in $S \setminus S_f$. Any model that exhibits this property cannot be from a model distribution produced by (re)training without S_f for all T', and has not unlearnt. Thus, passing our evaluation is necessary to claim an unlearning procedure can handle arbitrary deletions.

Comparable Across Training Procedures: Unlearning procedures often significantly modify the training procedure or architecture Bourtoule et al. (2021); He et al. (2021); Golatkar et al. (2020a,b, 2021); Graves et al. (2021). Thus, a versatile unlearning evaluation should provide measurements of retained information that are comparable across chanes in architectures and training procedures. For example, measuring relearn time as an evaluation method requires setting a threshold. However, different unlearning procedures (Golatkar et al., 2020a,b, 2021; Chundawat et al., 2023b) may differ in learning rate or have inherently different behavior in how low the loss can get and how fast it decreases. Similarly, L_2 distance between weights cannot be compared across architectures or hyperparameter choices like the amount of weight decay.

Checks Property Generalization: Unlearning procedures must ensure that properties which are only present in S_f do not influence performance on unseen samples. Some evaluations, such as membership inference attacks (MIA) (Shokri et al., 2017; Chen et al., 2021) effectively only determine the removal of memorization, and not the removal of generalized properties. In any evaluation with I.I.D removal, while it is theoretically possible to check if generalized properties are removed, it is not clear what properties to look for.

Two relevant properties that can be exacerbated by corrupted data are adversarial error (R_{adv}) (Madry et al., 2018) and unfairness[3] (Γ). An evaluation method that can be satisfied without removing these properties is clearly insufficient to guarantee unlearning. Theorem 1 shows that metrics like L_2 distance

[3]We use accuracy discrepancy (Buolamwini and Gebru, 2018; Sanyal et al., 2022) for mathematical simplicity. Can be shown for demographic parity and equalised odds (Hardt et al., 2016).

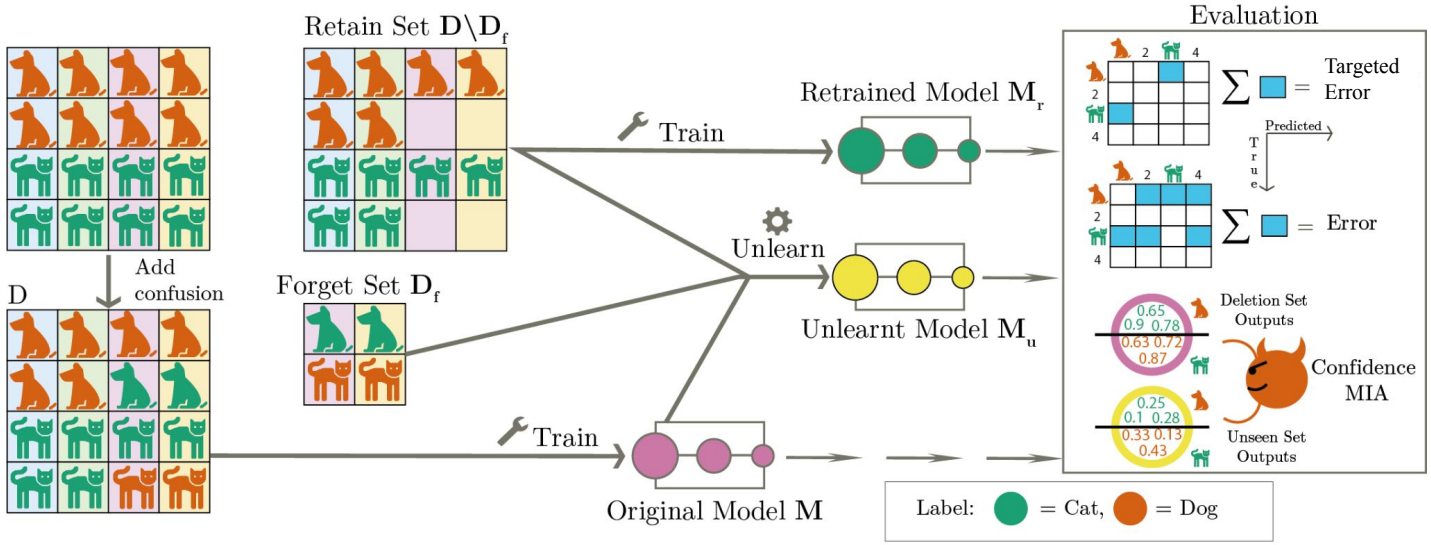

Figure 2.1: IC Test Pipeline: We mislabel a subset of samples from two classes of the original dataset, forming S_f. Here, shape and colour represent the actual and labelled class respectively. Then, M and M_r are obtained by training from scratch on S and $S \setminus S_f$ respectively. The unlearning procedure can leverage (some of) M, S_f and $S \setminus S_f$ to produce the unlearnt model M_u.

in the parameter space and gap in test error on a random I.I.D sample (R) are poor indicators of whether two models have similar R_{Adv} and Γ.

Theorem 1. *There exists a distribution $\mathcal{D}$ such that for any $\epsilon, \alpha \geq 0$, there exist two ℓ-layered fully connected linear NNs parameterised by $\mathcal{W}_1, \mathcal{W}_2$ which are simultaneously:*

- ***Close in Weights**:* $\|\mathcal{W}_1 - \mathcal{W}_2\|_F \leq \epsilon$
- ***Close in Test Error**:* $R\left(f_{\mathcal{W}_1}\right) \leq R\left(f_{\mathcal{W}_2}\right) + \alpha$
- ***Far in Robustness**:* $R_{\mathrm{adv}}\left(f_{\mathcal{W}_1}\right) \geq R_{\mathrm{adv}}\left(f_{\mathcal{W}_2}\right) + 1 - 2\alpha$
- ***Far in Fairness**:* $\Gamma\left(f_{\mathcal{W}_1}\right) = \Gamma\left(f_{\mathcal{W}_2}\right) - 1$

where $R, R_{\mathrm{Adv},\Gamma}$ are as defined above and $f_{\mathcal{W}}$ is an ℓ-layered fully connected linear neural network parameterised by W.

Proof. We prove this by constructing two ℓ-layered fully connected linear NNs, parameterised by $\mathcal{W}_1, \mathcal{W}_2$ and a distribution $\mathbb{P}$ such that, under $\mathbb{P}$ they are close in weights and test error but far in robustness and fairness.

Let $\mathcal{W}_1 = \{A^1, \ldots, A^\ell\}$ and $\mathcal{W}_2 = \{B^1, \ldots, B^\ell\}$ be the list of weight matrices of the two networks with each matrix having a dimension of $m \times m$. Consider all but the first layer of the two networks be identical. Specifically,

$$A^2 = A^3 \cdots = A^l = \begin{bmatrix} \sigma & & & & \\ & \sigma & & & \\ & & \ddots & & \\ & & & \sigma & \\ & & & & \sigma_1 \end{bmatrix}$$

with the remaining entries being 0 where we will define σ, σ_1 later. Construct the first layer of the two networks as follows where $\epsilon > 0$.

$$A^1 = \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & 0 \end{bmatrix} \qquad B^1 = \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & \epsilon \end{bmatrix}$$

Closeness in L_2 weights By construction, the two neural networks are close in weights: $\|\mathcal{W}_1 - \mathcal{W}_2\|_1 = \epsilon$ where $\|\mathcal{W}_1 - \mathcal{W}_2\|_1 = \sum_{i=1}^{\ell} \left\|\mathrm{A}^{\mathrm{i}} - \mathrm{B}^{\mathrm{i}}\right\|$.

To complete the remainder of the proof, note that the two neural networks are essentially equivalent to linear functions with the weight parameters A and B respectively where

$$A = \begin{bmatrix} \sigma^{l-1} & & & & \\ & \sigma^{l-1} & & & \\ & & \ddots & & \\ & & & \sigma^{l-1} & \\ & & & & 0 \end{bmatrix} \qquad B = \begin{bmatrix} \sigma^{l-1} & & & & \\ & \sigma^{l-1} & & & \\ & & \ddots & & \\ & & & \sigma^{l-1} & \\ & & & & \sigma_1^{l-1}\epsilon \end{bmatrix}$$

Next, we construct a data distribution $\mathbb{P}$ that satisfies the criteria of our result. Our distribution $\mathbb{P}$ will be supported on four points $X^1, X^2, X^3, X^4 \in \mathbb{R}^m$ where

$$X^1 = \left(\underbrace{1, \ldots, 1}_{m-1}, 0\right), X^2 = \left(\underbrace{-1 \ldots, -1}_{m-1}, 0\right), X^3 = \left(\underbrace{1, \ldots, 1}_{m-1}, -1\right), X^4 = \left(\underbrace{-1, \ldots, -1}_{m-1}, 1\right)$$

and $\mathbb{P}$ is defined as

$$\mathbb{P}[(X^1, +1)] + \mathbb{P}[(X^2, -1)] = 1 - \alpha \text{ and } \mathbb{P}[(X^3, +1)] + \mathbb{P}[(X^4, -1)] = \alpha.$$

Test Error It is easy to verify that if $\sigma > 0$, then $R(f_{\mathcal{W}_1}), R(f_{\mathcal{W}_2}) \leq \alpha$.

Fairness Now, let $\left\{X^3 \cup X^4\right\}$ be the minority group and $\left\{X^1 \cup X^2\right\}$ be the majority group. Note that $f_{\mathcal{W}_1}\left(X^1\right) = f_{\mathcal{W}_1}\left(X^3\right) = 1$ and $f_{\mathcal{W}_1}\left(X^2\right) = f_{\mathcal{W}_1}\left(X^4\right) = -1$, thereby leading to $\Gamma(f_{\mathcal{W}_1}) = 0$. On the other hand, for any ϵ, m if σ, σ_1 are chosen such that

$$\sigma_1^{l-1}\epsilon > (m-1)\sigma^{l-1}, \tag{2.1}$$

we have that $f_{\mathcal{W}_2}\left(X^3\right) = -1$ and $f_{\mathcal{W}_2}\left(X^4\right) = 1$. Hence, $\Gamma\left(f_{\mathcal{W}_2}\right) = 1$. This completes the proof of

$$\Gamma\left(f_{\mathcal{W}_2}\right) - \Gamma\left(f_{\mathcal{W}_1}\right) = 1.$$

Adversarial Robustness Let $\delta > 0$ be the adversarial perturbation budget. Then, the adversarial error of a network parameterised with parameters $\mathcal{W}$ is

$$\begin{aligned} R_{\mathrm{Adv}}(f_{\mathcal{W}}) &= \mathbb{P}_{X,y}\left[\exists \mathbf{z} \in \mathbb{R}^m \text{ s.t. } \|\mathbf{z}\| \leq \delta \ \wedge \ f_{\mathcal{W}}\left(X + \mathbf{z}\right) \neq y\right] \\ &\geq \alpha \mathbb{I}\left\{\exists \mathbf{z}_1, \mathbf{z}_2 \in \mathbb{R}^m \text{ s.t. } \|\mathbf{z}_1\|, \|\mathbf{z}_2\| \leq \delta \ \wedge \ f_{\mathcal{W}}\left(X^1 + \mathbf{z}_1\right) \neq 1 \ \wedge \ f_{\mathcal{W}}\left(X^2 + \mathbf{z}_2\right) \neq -1\right\}. \end{aligned}$$

Note that if the parameters σ, σ_1 satisfy the following with respect to ϵ, δ

$$\sigma_1^{l-1}\epsilon\delta > (m-1)\sigma^{l-1} \tag{2.2}$$

then $f_{\mathcal{W}_2}\left(X^1 - \delta e_m\right) = -1$ and $f_{\mathcal{W}_2}\left(X^2 + \delta e_m\right) = 1$ where e_m is the m^{th} canonical basis vector. Thus, $R_{\text{Adv}}(f_{\mathcal{W}_2}) \geq 1 - \alpha$. It is also easy to verify that for any $\mathbf{z} : \|\mathbf{z}\| \leq \frac{1}{2}$, we have that for $f_{\mathcal{W}_1}\left(X^1 + \mathbf{z}\right) = 1$ and $f_{\mathcal{W}_1}\left(X^2 + \mathbf{z}\right) = -1$. Thus, $R_{\text{Adv}}(f_{\mathcal{W}_1}) \leq \alpha$. Subtracting the two adversarial errors we obtain, $R_{\text{Adv}}(f_{\mathcal{W}_1}) - R_{\text{Adv}}(f_{\mathcal{W}_2}) \geq 1 - 2\alpha$.
Finally, combining Equations (2.1) and (2.2) and setting the parameters such that $\frac{\sigma_1}{\sigma} \geq \left(\frac{m-1}{\epsilon\delta}\right)^{\frac{1}{l-1}}$ completes the proof. □

The theorem shows that two models that are arbitrarily close in weights and test error can be arbitrarily far in adversarial robustness and fairness. In particular, we show an example where models get farther in adversarial robustness as they get closer in test error. Thus, unlearning evaluations must either measure indistinguishability in terms of more *adversarial* quantities like robustness and fairness or use strategic non-I.I.D deletion sets. In our work, we explore the latter, i.e. an adversarial approach to designing deletion sets. In the following section, we introduce this evaluation procedure known as IC test.

2.2.3 Proposal: Interclass Confusion Test

In contrast to existing evaluations, we inject a strong differentiating influence specific to S_f into the training dataset via label manipulations. Specifically, we present:

Interclass Confusion (IC): As illustrated in Figure 2.1, the IC test using a deletion set of n samples follows these steps:

1. Take $\frac{n}{2}$ samples each from two classes in the train data to form $S' \subset S$ (Targeted sampling).
2. Swap labels [4] between the two classes of samples in S' (Adversarial manipulation) to get the confused set S_f. The dataset for training the original model M is $(S \setminus S') \cup S_f$.
3. Select the set S_f as data to be deleted from the trained model M (Strategic deletion set).
4. Evaluate memorization and property generalization by measuring error on training and testing sets S' and S'_u corresponding to the two classes.

To isolate the effect of targeted sampling in the IC test *i.e.*confusing two specific classes, we introduce:

Ablation: I.I.D Confusion: We select n samples uniformly at random from S to form S' and mislabel them to a uniform random different class, using these mislabelled samples as S_f. Note that the removal is not I.I.D, we replace targeted label manipulation with I.I.D label noise.

[4]Note that evaluations based on label swapping have been used in traditional adversarial robustness literature (Nakkiran, 2019; Fowl et al., 2021), but with quite different goals, setting and design.

We compute the MIA and Error on affected classes like previous work, but also introduce the Targeted Error metric:

Error v/s Targeted Error: Error computed for a given set S is the fraction of samples in S which were misclassified regardless of which class it was mistaken as. In Interclass Confusion, we are interested specifically in the fraction of samples confused between the two confused classes. In Class Removal, we are interested in the fraction of samples classified as the class to be removed. This is measured by Targeted Error, which is the fraction of samples in S misclassified to the targeted class exhibiting the unwanted property is not removed. Samples misclassified into any other class are not counted as illustrated in Figure 2.1 for IC test. As an illustrative example, for IC test on a 10 class dataset: the error of a random model would be 90%, but the targeted error would be 10%. Error/Targeted Error when computed on the set S' measures memorization, and when computed on the unseen (test) set samples S_t from the same distribution as set S' measures property generalization.

2.3 Unlearning Baselines

Having discussed properties of evaluation methods, we now discuss unlearning procedures – desirable properties and our two simple baselines that achieve them.

2.3.1 Desiderata for Unlearning Methods

Unlearning procedures need the ability to learn: Consider a linearly-separable binary classification task where we use the IC test to introduce complete confusion between the two classes (50% of samples of each class mislabelled as the other). Powerful empirical risk minimizers (like neural networks trained with SGD) will achieve a train accuracy on S close to 100% (Zhang et al., 2017). However the test accuracy will be much lower, closer to 50%, as the training dataset is essentially fully randomly labelled. However, upon deleting all the mislabelled samples, like in the IC test, we are left with 50% of the original dataset but with correct labels. A model retrained from scratch on $S \setminus S_f$ can be expected to achieve reasonably good accuracy, much larger than 50%, which a good unlearning procedure is expected to match. So we can expect the unlearnt model to have learnt to perform the task, whereas the original model cannot.

Intuitively, this implies that *solely erasing information from the model is not enough, and the ability to learn may be necessary for ideal unlearning procedures*. Consequently, we expect methods which do not use information about the retain set (Chundawat et al., 2023b) will have limitations when handling arbitrary deletions and will not perform well on the IC test.

Scalability to large deletion sets: Popular unlearning methods, both exact and inexact, either explicitly assume tiny deletion sets (Thudi et al., 2021; Bourtoule et al., 2021; Wu et al., 2020) or scale poorly beyond them in practice (Schelter, 2020; Graves et al., 2021; Golatkar et al., 2020a,b). In Section 2.5.1 we show that the computational complexity of methods based on the paradigm of isolating the influence of data to small parts of the training procedure (Bourtoule et al., 2021; He et al., 2021; Yan

et al., 2022; Graves et al., 2021) scales exponentially with the size of the deletion set. Arguably, methods which require resources similar to retraining from scratch for large deletion sets have limited practical value, especially in applications which require large deletion sets (see Section 2.5.1 for a discussion).

Targeting Areas to Unlearn: A way to significantly improve efficiency of unlearning procedures is to focus optimization power towards areas of a model where the deletion set is stored. We look from a layerwise perspective- the early layers of a deep network capture generic low-level representation (Yosinski et al., 2015; Kataoka et al., 2020), while the later layers focus on dataset-specific information. Interestingly, the earlier layers are also the most computationally intensive (Brock et al., 2017). Hence, focusing unlearning on the last k layers may allow computationally efficient erasure of information from S_f. Such unlearning methods also help us analyze how early in a deep network can an evaluation method detect the presence of information specific to S_f.

Overall, unlearning methods should: (i) have capacity to learn information in addition to unlearning and (ii) scale to large deletion sets and further, for our analysis, we wish to have unlearning methods that (iii) target specific parts of the model, e.g. the last k layers for unlearning.

2.3.2 Proposal for Unlearning baselines: CF-k and EU-k

We propose two methods which we believe will be useful 'baselines' for future work to compare against. (i) They achieve a tradeoff between forgetting and efficiency which can be controlled using parameter k, allowing comparisons with unlearning procedures of differing degrees of efficacy. (ii) They are simple and require minimal assumptions: they scale to large deletion sets, are applicable for all DNN training procedures and require only access to $S \setminus S_f$.

Exact-unlearning the last k layers (EU-k): We retrain the last k layers of M from scratch using the same training procedure T on retain set $S \setminus S_f$ while freezing prior layers.

Catastrophically forgetting the last k layers (CF-k): Neural Networks suffer from catastrophic-forgetting (French, 1999) - when a model is continually updated without some previously learnt samples, the model loses knowledge about them. We finetune the last k layers of M on the retain set $S \setminus S_f$ using the same training procedure T while freezing prior layers, hoping to catastrophically forget S_f. As we avoid re-initializing the last k layers unlike EU-k, we need far fewer epochs, making CF-k more efficient than EU-k.

2.4 Experiments

We show empirical support for three claims of our work. (i) We show that our EU-k and CF-k unlearn better than four popular methods and are strong baselines. (ii) Using EU-k and CF-k for analysis, we show our primary contribution, the IC test, is more reliable than previous evaluations in detecting unwanted memorization and property generalization. (iii) We show standard regularization techniques can make original models M more amenable to unlearning. Our training procedure is described in

Section 2.5.3.1. All code has been made publically available at https://github.com/shash42/Evaluating-Inexact-Unlearning.

2.4.1 Evaluating Baselines: EU-k and CF-k

Setup. Due to the lack of established evaluation methods and comparisons to other methods in past work, the 'state of the art' in unlearning is not clear. We compare our proposed baselines against four popular past unlearning methods: Fisher (Golatkar et al., 2020a), NTK-Fisher (Golatkar et al., 2020b), Amnesiac Unlearning (Graves et al., 2021), and LCODEC (Mehta et al., 2022) which have published their codebase for accurate reproduction. We could not run Fisher, NTK-Fisher for our larger datasets like CIFAR10 due to large memory requirements and thus compare all models on their setting: We use Small-CIFAR-5 (Golatkar et al., 2020a) (a 5 class subset of CIFAR10), and all samples of a given class as the S_f (Class Removal). We follow their training procedure to get their original and retrain models, as we obtained near-random performance when we applied their unlearning method on our standard training procedure perhaps due to violation of some of their training assumptions. For Amnesiac, LCODEC and our unlearning procedures we report results on unlearning from an original model M produced by our default procedure (T) with a standard ResNet-20 architecture. We obtain the same observations on using their respective training procedures which produce original models with lower accuracy. For forgetting we report memorization and property generalization by computing targeted error on the deletion set (S_f) and test set of the deleted class (S_t) respectively. We measure accuracy with test set error and efficiency with unlearning time.

Results. We present all our results in Table 2.2.

Accuracy: The test error of the retrained model is 15% higher than the original (both T) because a portion of the test samples belong to the deleted class. Here, lower test errors are attributable to not forgetting the deleted class. Comparing test error for the Original models, we observe our procedure T has a large decrease (10%) in test error compared to Golatkar et al. (2020b). This ensures we study unlearning on better, more realistic models. We find that Amnesiac and LCODEC produce unlearnt models with almost random performance. Amnesiac relies on deletion set samples belonging to only a few batches. However, this assumption does not scale to large deletion sets and we find all batches are affected in our experiment, as expected from the mathematical analysis we present in Section 2.5.1. LCODEC removes samples sequentially, and the error of the model increases fast as more samples are deleted.

Forgetting: We measure the degree of unlearning of a given method by comparing the reduction in targeted error of the method with the corresponding original and retrained models providing the starting and ideal scores respectively. We observe that simply unlearning the last layer with our baselines (EU-1 & CF-1) have far better reductions in targeted error compared to previous methods in both memorization and property generalization. Surprisingly, Golatkar et al. (2020a) fails to achieve any significant forgetting.

Efficiency: We observe that 3 out of 4 past procedures take far more time for unlearning compared to our baselines and even retraining as approximating the Fisher Information Matrix is expensive. In

Table 2.2: Comparison between unlearning procedures on Class removal test on Small-CIFAR-5. Forgetting measured by targeted error: Memorization (Mem) and Property Generalization (PropGen). Performance and efficiency measured by test error and unlearning time. For all metrics, lower is better.

Model		Targeted Error		Test Error	Time(s)
		Mem	PropGen		
T from Golatkar et al. (2020a,b)					
Original		92.3	97.6	26.7	0.00
Fisher Golatkar et al. (2020a)		94.6	98.0	33.2	141.95
NTK-Fisher Golatkar et al. (2020b)		27.0	39.6	31.0	141.90
Retrain		0.0	0.0	41.4	9.81
T (Ours)					
Original		98.0	97.3	16.3	0.00
Amnesiac (Graves et al., 2021)		22.3	21.6	74.3	1.72
LCODEC (Mehta et al., 2022)		20.7	20.2	80.3	226.9
1-layer (Ours)	CF	18.3	12.3	30.9	4.43
	EU	9.6	4.3	31.9	9.38
10-layers (Ours)	CF	15.6	9.3	29.4	5.22
	EU	2.0	0.0	32.6	10.78
Retrain		0.0	0.0	32.5	12.33

real-world scenarios, such speedups are highly important to enable practical applications of unlearning. Amnesiac is fast but produces a random model.

Conclusion. Our methods EU-k and CF-k outperform popular unlearning methods by significant margins in all three dimensions: forgetting, accuracy and efficiency indicating they are reasonable baselines for analysis.

2.4.2 Comparing Tests for Evaluating Forgetting

Setup. We use CIFAR10 and CIFAR100 datasets with a 40K-10K-10K train-val-test split. Note that we use the same deletion set size n for a fair comparison across all tests, with the sample set removed for every test, with details in Section 2.5.3.3. Experiments in this section use n corresponding to the number of training samples in one class: 4000 for CIFAR10 and 400 for CIFAR100 (Krizhevsky et al., 2009). We further report results across different deletion set sizes n in the Section 2.6.3, 2.6.4 and find them to be consistent. All results are averaged over three runs with different seeds for robustness.

Results. In Figure 2.2 we compare different unlearning evaluation methods on their ability to demonstrate the degree of forgetting of models produced by our baselines EU-k and CF-k. Every line

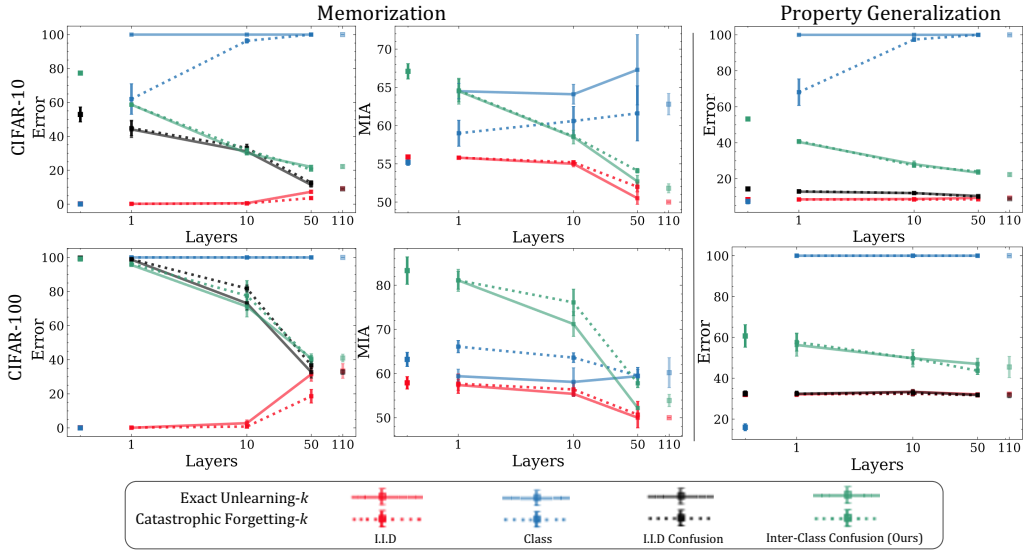

Figure 2.2: Error, MIA for various deletion strategies (Y) reported across the number of layers (X) affected by the unlearning procedure. The left-most points at 0 layers represent the original model M, whereas the right-most points at 110 layers represent the retrained model M_r^T. Only Interclass Confusion reliably distinguishes different degrees of unlearning (no. of layers unlearnt) across all graphs.

is formed by varying the number of layers unlearnt k and hence the degree of forgetting, with 0 and 110 (leftmost and the rightmost points) indicating the original and retrain models respectively. A strong test is indicated by: (i) the score of intermediate models ($0 < k < 110$) is different from that of the retrain model as some information is still retained after unlearning k layers. (ii) There is a clear gradual improvement in the forgetting metric as k increases. We present results consistent across graphs below:

Memorization: Across settings, class removal test (blue) is not able to detect memorized information even in simply exact-unlearning the last layer on any metric or dataset (solid-blue line reaches retrain scores immediately). I.I.D removal barely distinguishes 1-layer and 10-layer unlearning. We get random (<50%) MIA scores for I.I.D confusion and hence exclude it. However, I.I.D confusion performs as well as the IC test on the error metric. IC test is the most useful across metrics and datasets, clearly distinguishing models with information removed from more layers.

Property Generalization: Only IC test is capable of detecting property generalization of confusion even after exact unlearning just the last layer. Even I.I.D confusion, which represents adding noisy labels with no systematic bias, is clearly insufficient to induce detectable generalized properties. Thus, both components of the IC test, class-targeted removal and confusion, are needed together to show clear trends in property generalization evaluations.

Hyperparameters of IC Test: The two hyperparameters in executing the IC test are choosing the two classes to confuse and the number of confused and deleted samples n. We find that while trends are similar across class pairs, unlearning is the hardest when we choose classes that are highly similar. We

thus report results for (Cat, Dog) in CIFAR 10 and (Maple Tree, Oak Tree) in CIFAR 100 here. Further details can be found in Section 2.6.5. Regarding the size of the deletion set, we demonstrate that the IC test can reliably detect imperfect memorization and property generalization with just 1% and 5% of the dataset being corrupted respectively. We report results for 5% here, and show IC test is the most useful among all tests across deletion set sizes in Section 2.6.4, 2.6.3.

EU-k v/s CF-k: For any given k, catastrophic-forgetting (CF) removes most of the information in those layers while being twice as fast as exact-unlearning (EU) indicated by the dotted lines closely following their solid counterparts across tests. The only cases where they differ is when the test is simply unable to detect information retained in exact unlearning (e.g. Class Removal). As shown by the IC test, EU-1 leaks a lot of information gained from S_f, showing that prior exact unlearning methods that only modify the final layer of deep networks (Baumhauer et al., 2022; Izzo et al., 2021) continue to store information from D_f and cannot handle arbitrary deletions [5]. Note that EU-k and CF-k continue to maintain the same accuracy across k as shown in Table 2.3. While we choose a 110 layer ResNet and $k = \{1, 10, 50\}$ as a concise representative sample here, all observations hold across more values of k and network depths if compared using the fraction of layers unlearnt as shown in Section 2.6.2.

Conclusion. IC test is the only test that shows a clear difference between models with different number of layers unlearnt for both memorization and property generalizaton, on all metrics and datasets. IC test also shows past unlearning methods that propose to modify only the final layer of deep networks continue to retain most information about the deletion set. Varying k in EU-k and CF-k can be used to control the forgetting-efficiency tradeoff at the same overall accuracy. Catastrophic forgetting achieved similar degree of forgetting as exact unlearning, while being twice as efficient.

2.4.3 Making Models Amenable to Unlearning

Aim. Different original models M can have varying propensities to memorize D_f. We aim to leverage this to provide training strategies that obtain original models M with better unlearning properties, particularly computational efficiency. This is in line with Thudi et al. (2022) which theoretically motivates this for $|S_f| = 1$, but we empirically show it holds even for large deletion sets.

Strategies. Early stopping has been a universal strategy to prevent overfitting (*i.e.*memorization) in machine learning. We also use Cutmix (Yun et al., 2019), with the intuition that the model never sees a training sample in isolation while training, inspired from Huang et al. (2020b). Apart from using these regularization strategies during training, we use the same setup as before.

Results. We present results in Figure 2.3. Comparing original models (leftmost group of bars on the graphs), we observe that both techniques obtain large reductions in memorization of S_f but similar property generalization. We observe only a marginal dropoff in unlearning (especially property generalization) from Cutmix+Early Stopping 10 layers to Original 50 layers. Cutmix+Early Stop 10 layers gives a huge improvement in unlearning performance compared to the Original 10 layers unlearnt

[5] Only unlearning the final layer may succeed if earlier layers are trained privately (Guo et al., 2020; Wu et al., 2020).

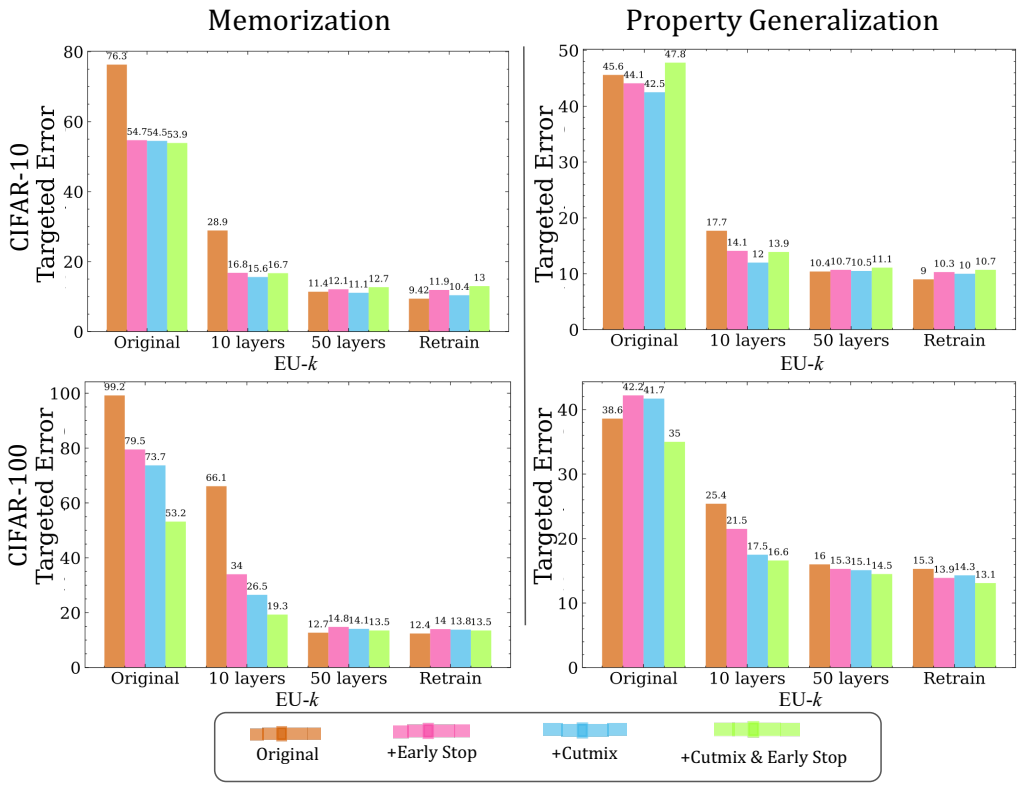

Figure 2.3: Interclass Confusion Targeted Error (Y) on unlearning from original models with different regularization (bar colors) reported for the original model M, EU-10, EU-50, and retrained model M_r^t. The same unlearning procedure can remove more confusion when starting from better regularized original models.

models, especially on the harder CIFAR-100 dataset. In property generalization, this occurs despite original models having similar amounts of confusion indicating better regularized models make it easier for inexact unlearning methods to remove information.

Conclusion. The presented results validate the idea that some original models make it easier to remove information using the same unlearning procedure. We demonstrated how this can be leveraged to achieve forgetting using cheaper unlearning procedures. Comparisons across unlearning procedures should ideally use the same original model for fairness, at least when there are no training assumptions.

2.5 Design Choices

We provide details of implementation, test and metric choices, unlearning method comparisons and utility calculations.

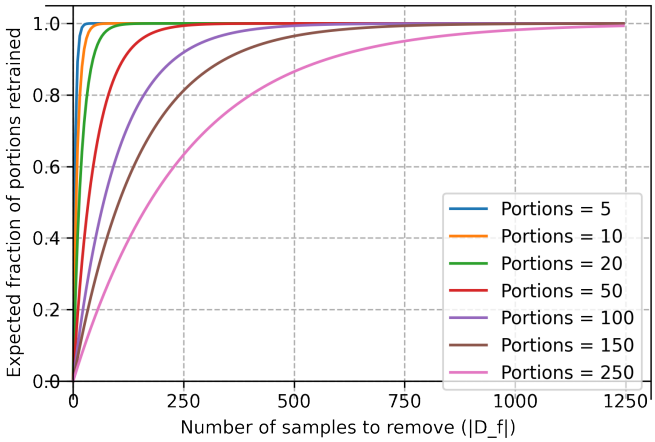

Figure 2.4: Hyperbolic deterioration of efficiency in isolation-based unlearning when scaling to a large number of removed samples. In this work, we analyze $|S_f|$ from 100-4000 where $\mathbb{E}[Y] \sim 1$.

2.5.1 Against Isolation Strategies

Examples include removing noisy labels (Northcutt et al., 2021b,a), deleting poisoned samples (Wang et al., 2019; Jagielski et al., 2018; Li et al., 2020), deleting data that induces harmful biases (Prabhu and Birhane, 2021; Fabbrizzi et al., 2021), and organizations requiring deletion of user data older than some retention period. Even in the context of privacy, a single user might *own* multiple samples in the dataset. In biometrics like face recognition (Turk and Pentland, 1991), one user may form an entire class (Baumhauer et al., 2022). Moreover, user deletion requests may occur in bursts after certain *events of interest*, such as revelations of privacy leakages by an organization (Acquisti et al., 2006). Lastly, batching online deletion requests requires less invocations of the unlearning procedure, boosting resource efficiency.

A popular approach for unlearning is data-influence isolation, where each sample is made to contribute only to a small part of the training procedure or model. Unlearning such as retraining from scratch only the part affected by the deletion set erases the influence of the deletion set more efficiently. Isolation-based strategies change the training process by creating an ensemble (Yan et al., 2022; Schelter, 2020; Bourtoule et al., 2021; Graves et al., 2021; He et al., 2021), each of whose models is trained on different subsets of the dataset. This ensures architecturally (Bourtoule et al., 2021; Aldaghri et al., 2021; Schelter, 2020; Yan et al., 2022) or temporally (He et al., 2021; Bourtoule et al., 2021) isolating the influence of any sample to a limited part of training, requiring retraining for only the affected parts. Isolation has been used across techniques like Linear Classification (Aldaghri et al., 2021), Random Forest (Schelter et al., 2021; Brophy and Lowd, 2021), KNN (COO, 1982), SVM (Cauwenberghs and Poggio, 2000; Tsai et al., 2014) and DNN (Graves et al., 2021; Bourtoule et al., 2021; He et al., 2021) by utilizing or creating a sparse influence graph (Schelter, 2020). Data-influence isolation often comes at the cost of utility as each portion becomes a weaker learner (Banko and Brill, 2001), especially in deep networks (Shorten and

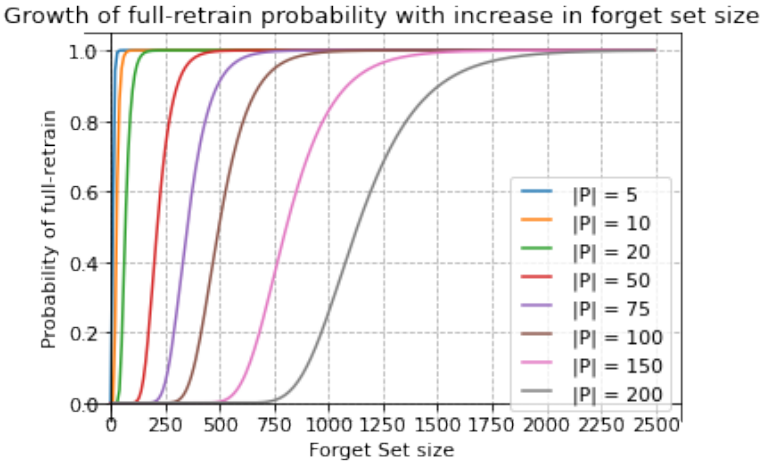

Figure 2.5: Logistic growth of the probability of needing to retrain all portions with increasing deletion set size. We represent isolation strategies with different portion sizes P.

Khoshgoftaar, 2019). To overcome the dropping utility, the training and unlearning time may need to be increased, reducing resource efficiency.

Figure 2.4 demonstrates that the computation costs of isolation-based strategies scale poorly as the deletion set size increases. Note that even on a practical deletion set size like 500, existing isolation based approaches (which create much less than 250 isolated portions) require almost full retraining costs on expectation. Let P be the number of parts obtained with the isolation strategy. We assume the best-case scenario where each sample only influences one part. We make the simplifying assumption that the samples are uniformly distributed across parts, and the probability of a removed sample belonging to any particular portion remains constant ($\frac{1}{P}$). Let Y be the number of affected parts. The probability part i is affected by atleast one sample in S_f is $1-(1-\frac{1}{P})^{|S_f|}$. Thus by the linearity of expectation: $\mathbb{E}[Y] = P\left(1-\left(1-\frac{1}{P}\right)^{|S_f|}\right)$.

We also show that the probability of full-retrain in data-influence isolation unlearning methods scales poorly with increasing deletion set size. Let $p(n)$ be the probability that all P portions are affected on the removal of n samples. Extending the analysis of Warnecke et al. (2021) from the specific case of SISA to data-influence isolation in general, we get:

$$p(n) = 1 - \frac{\sum_{j=1}^{|P|}(-1)^{j+1}\binom{|P|}{j}(|P|-j)^n}{|P|^n}$$

Figure 2.5 shows $p(|S_f|)$ grows logistically, implying there is a fast increase in the chance of needing a full-retrain as deletion sets get larger. This demonstrates how data-influence isolation provides little improvement in efficiency compared to the retrain-from-scratch baseline for practical scenarios.

Note that some prior work such as Graves et al. (2021) do not re-train the affected portions without the deleted data, instead removing them entirely. This replaces the hit on resource efficiency with decreased utility (such as accuracy) as more deletions lead to more affected portions being removed from the model.

This explains why Graves et al. (2021)'s method produces an almost random model in the Class Removal experiment shown in Table 2.2.

2.5.2 Membership Inference Attacks

Background Membership Inference Attacks (MIA) (Shokri et al., 2017) can be used to determine whether a particular sample was part of the training data of a model. Many different black-box formulations of MIA have been used to measure the efficacy of unlearning. Most (Golatkar et al., 2020b, 2021; Ma et al., 2023; Graves et al., 2021) learn a binary attack classifier: based on the model's output for the sample, was the sample in the seen training set (class 0) or the unseen test set (class 1)? The attack classifier is then applied on deletion set samples, with ideal unlearning entailing all samples are classified as unseen. However, such a test is extremely sensitive to the efficacy of the attack classifier which may be unreliable. Another approach has been to train the attack classifier to distinguish the outputs of a large number of original (M) and retrained (M_r^T) models and then classify the unlearnt model M_u (Baumhauer et al., 2022). This formulation involves prohibitive computational expense and still can't check over all potential $T' \neq T$, indistinguishability with any of whom would guarantee unlearning.

Song and Mittal (2021) show that *metric-MIA*, measuring simple metrics and deciding membership based on a threshold, can match the classification accuracy of trained attack models. In particular, their confidence-based MIA measures the model's output probability for the target class and selecting separate class-wise membership thresholds. It is shown to match the performance of even white-box MIA attack classifiers.

Our Formulation

We adapt the confidence-based MIA (Song and Mittal, 2021) to propose an efficient black-box MIA formulation specifically tailored for measuring forgetting. We assume direct access to the actual model outputs instead of shadow models (Graves et al., 2021; Ma et al., 2023), as shadow models only weaken the attack, making the unlearning test artificially easier to pass. We distinguish the model outputs on samples from the deletion set S_f and unseen samples S_t from the same underlying distribution rather than training an attack classifier using the entire train and test set. We believe this formulation is a more targeted measurement of forgetting as it directly discriminates between outputs on S_f and S_t in contrast to train and test set used in past literature (Golatkar et al., 2020b, 2021; Graves et al., 2021; Ma et al., 2023).

Our MIA takes in model M, forget set S_f and unseen samples S_u from the same classes found in S_f. The following procedure is repeated for each 'target class' t:

- Dataset S_{MIA} is created with the probability outputs for class t: $M(S_f)_t$ and $M(D_u)_t$ stored as class 0 and class 1 respectively.
- We then create a 50-50[6] shadow (S_{MIA-S}) - test (S_{MIA-T}) split of S_{MIA}.

[6]Given that only 1 parameter (threshold) needs to be learnt, the shadow size is sufficient

- A threshold p_t needs to be chosen such that probabilities $> p_t$ are classified as class 0, and probabilities $< p_t$ as class 1. The p_t that maximizes the accuracy on S_{MIA-S} is chosen.
- The accuracy obtained on S_{MIA-T} using threshold p_t is the MIA accuracy for target class t. A weighted average of this test accuracy across all target classes is taken as the final MIA accuracy.

Usually, the target class t is the actual label of the sample. However, in the case of IC test, we use the mislabelled class as the target for both, S_f and S_u samples. Intuitively, the memorization of mislabels in the deletion set would make the wrong class probability output unnaturally higher than other unseen samples of the same class, making the MIA stronger. Such an enhancement is not possible in the case of I.I.D confusion as the mislabels are untargeted.

In line with existing MIA literature, we want our attack classifier accuracy to be 50% incase of no classifier advantage. Thus as the forget set and unseen set may have differing sizes in some experiments, we take a random subset of the larger one to make the attack dataset balanced. The numbers reported are averaged over 20 runs with randomness induced by the subset sampling step. Note that since the classifier learns to distinguish between the test and forget set distribution directly, it might be able to distinguish them spuriously, leading to slightly more than 50% attack classifier accuracy even on perfect unlearning. Thus, the reference gold standard MIA performance can instead be that of any exactly unlearnt model upon undergoing the same evaluation.

2.5.3 Hyperparameters

We now provide some additional details for results shown in the main paper.

2.5.3.1 Implementation Details

Training. We use the ResNet architecture (He et al., 2016) with 110 layers. Our standard training procedure T is as follows: We train our models for 62 epochs (CIFAR10) or 126 epochs (CIFAR100), using a SGD optimizer with momentum 0.9 and weight decay 5e-5, an SGDR scheduler with t_{mult} = 2, t_0 = 1, minlr = 5e-3, maxlr = 0.01 and a batch size of 64. For EU-k and CF-k baselines, we use this same training process, but on the final k layers. In CF-k, the only difference is we finetune for only half the epochs.

The setup used for all experiments is a PC with a Intel(R) Xeon(R) E5-2640 2.40 GHz CPU, 128GB RAM and 1 GeForce RTX 2080 GPU.

We make the following deviations in our experiments:

- In Table 2.2 we make changes described in Section **??**.
- In Figure 2.3 and Table 2.4 we change the training procedure. When using cutmix regularization, we use $p = 0.5$ and $\alpha = 1.0$. For early stopping, we halve the number of epochs both while training the original/retrain models and also in the unlearning procedures.

- In Table 2.5 we vary the number of finetuning epochs in CF-k.
- In Figure 2.11 we vary the confused classes in the IC test from easy-hard on the axis of distinguishability.
- In Figure 2.6 we further benchmark on ResNet-20, ResNet-56 and ResNet-110 to show our results are robust to the choice of network depths.
- In Table 2.6 we ablate the effect of warm restarts in training the original/retrain model.

2.5.3.2 Metrics

Inclusion in the Evaluation Comparisons Table Note that the list of metrics in Table 1 of the main paper does not include metrics like upper bound on information remaining in weights and activations (Golatkar et al., 2020a,b, 2021) since its unclear whether such metrics can be computed on methods other than their own proposed unlearning procedure. We also exclude purely-qualitative tests such as model inversion attacks (Fredrikson et al., 2015) which have been used in prior unlearning works (Graves et al., 2021; Baumhauer et al., 2022).

Details of Metric Computation

Targeted Error We propose Targeted Error which measures the number of samples classified according to a property (information) unique to S_f. For the IC-test, it is the fraction of samples still confused between the two classes, i.e. $\texttt{Targeted Error}(M, S, A, B) = \frac{C^{M,S}_{A,B}+C^{M,S}_{B,A}}{|S_A|+|S_B|}$. where $C^{M,S}$ is the confusion matrix when using model M outputs on dataset S and A, B are the classes confused. For confusion between $N > 2$ classes, targeted error is the sum of the confusion matrix terms for all pair-wise misclassifications among the N classes. Thus, targeted error converges to error when N is the same as the total number of classes, as in I.I.D Confusion. For Class Removal test, targeted error calculates the number of samples labelled as the removed class.

Note that the influence of utility on targeted error is significantly lesser than the simple error metric on the affected classes as illustrated in Figure 2.1. Regarding the passing score for IC test: We speculate achieving lower targeted error than randomly initialized models could be sufficient. However, achieving this score is not necessary: even on exact unlearning of S_f, our models obtain a higher score due to samples in the retained set ($S \setminus S_f$) having noisy annotations, an unavoidable phenomena in real-world datasets. We approximate this inherent noise in the dataset by using the model retrained from scratch.

For clarity, we further describe the computation of some metrics. Our MIA has already been described in Section 2.5.2. Note that for measuring memorization, the deletion set is used, while for measuring generalization (a subset of) the test set is used.

IC Targeted Error: For the IC test between class A and B, the targeted error represents the number of samples of class A mislabelled as class B and vice-versa. Intuitively, as the mislabelled samples are forgotten by the unlearning procedure, the model should confuse lesser samples between these two classes.

Method		I.I.D Removal (↓)	Class Removal (↓)	I.I.D confusion (↓)	IC (↓)
CIFAR-10 ($\|S_f\| = 4000$)					
Original		8.4 ± 0.2	8.8 ± 0.4	14.3 ± 0.7	6.9 ± 0.5
1-layer	CF	8.4 ± 0.1	8.3 ± 0.2	13.0 ± 0.9	6.5 ± 0.5
	EU	8.5 ± 0.1	8.4 ± 0.3	12.8 ± 0.9	6.6 ± 0.3
10-layers	CF	8.4 ± 0.2	8.4 ± 0.2	12.0 ± 0.6	6.5 ± 0.4
	EU	8.7 ± 0.1	8.6 ± 0.2	12.0 ± 0.7	6.7 ± 0.2
50-layers	CF	8.5 ± 0.1	8.1 ± 0.4	10.0 ± 0.4	6.1 ± 0.3
	EU	9.3 ± 0.3	8.8 ± 0.4	10.4 ± 0.4	6.9 ± 0.5
Retrain		9.3 ± 0.1	8.2 ± 0.3	8.8 ± 0.3	6.4 ± 0.2
CIFAR-100 ($\|S_f\| = 400$)					
Original		32.1 ± 1.1	31.6 ± 1.1	32.4 ± 1.4	31.8 ± 0.8
1-layer	CF	32.1 ± 1.0	31.7 ± 1.2	32.4 ± 1.3	31.7 ± 0.7
	EU	32.1 ± 1.0	31.7 ± 1.1	32.4 ± 1.2	31.8 ± 0.7
10-layers	CF	32.4 ± 0.9	32.2 ± 1.1	32.6 ± 1.3	32.0 ± 0.6
	EU	33.3 ± 1.2	32.5 ± 1.1	33.3 ± 1.2	32.8 ± 0.9
50-layers	CF	31.7 ± 0.9	31.5 ± 1.0	31.8 ± 0.8	31.2 ± 0.6
	EU	32.1 ± 0.3	31.6 ± 0.2	31.8 ± 1.0	31.7 ± 1.0
Retrain		32.2 ± 0.4	31.6 ± 0.6	31.7 ± 1.3	31.8 ± 1.0

Table 2.3: Error on the retain set distribution of test samples across unlearning tests. Scores are reported as: mean $\pm$ stdev. The EU-k and CF-k unlearning procedures lead to a minimal change in utility compared to retraining from scratch, unless utility is correlated with unlearning in the applied test.

Class Removal Targeted Error: For the class removal test removing samples from class A, the Targeted Error represents the number of samples the model classifies as class A. Intuitively, as more samples from A are removed, the model should classify lesser samples into A. Note that if the entire class is not removed, a model that generalizes better from the partial samples still available may get penalized unnecessarily.

IC Error: Error on train/test samples from the confused classes of the IC test, A and B.

Class Removal Error: Error on train/test samples of the removed class A.

I.I.D confusion, Error: Error on all samples from the train/test set. Here, a specific set of classes cannot be used for a targeted measurement.

2.5.3.3 Choices for Tests

In the IC test we confuse samples between classes 3 (Cat) and 5 (Dog) on CIFAR10 and classes 47 (Maple Tree) and 52 (Oak Tree) on CIFAR100 unless otherwise specified. Confusing two classes can

Method		None	Early Stop	Cutmix	Cutmix+Early
CIFAR-10 ($\|S_f\| = 4000$)					
Original		6.53	7.91	5.81	8.02
10-layers	CF	6.11	7.93	5.45	7.27
	EU	6.55	7.88	5.68	7.08
50-layers	CF	5.75	7.31	5.32	6.75
	EU	6.57	7.87	6.16	8.20
Retrain		6.31	8.50	5.70	8.97
CIFAR-100 ($\|S_f\| = 400$)					
Original		32.53	33.10	27.26	30.03
10-layers	CF	32.22	33.04	27.98	30.61
	EU	33.25	33.23	28.59	31.23
50-layers	CF	30.93	32.37	27.92	29.37
	EU	31.98	33.66	30.41	30.93
Retrain		30.64	32.62	26.67	30.67

Table 2.4: Error on the retain set distribution of test samples on varying the training procedure of the original model. Regularized models have better utility even after unlearning.

harm the overall accuracy of the original model, and we expect this effect to be more prominent when the total number of classes in the dataset is lower. The deletion set size is the same as the number of samples from one class in the training set unless otherwise specified. Note that while the size of S_f is the same when comparing different tests, the size of S_t is dependent on the test itself. In targeted tests (Class removal, IC), S_t only has test set samples from the affected classes, whereas in untargeted tests (I.I.D Removal, I.I.D confusion) S_t consists of the entire test set. In Class Removal test we remove class 0 for both CIFAR10 and CIFAR100, whereas in I.I.D Removal and I.I.D confusion we draw an equal number of samples randomly from each class.

2.5.3.4 Utilities

To measure utility, we compute error on unseen samples from the same distribution (unaffected classes) as $S \setminus S_f$, called the retain distribution. For the I.I.D Removal and I.I.D confusion tests, as the removal is untargeted, the evaluated samples are the same as the full test set. For the Class Removal and I.I.D confusion tests the evaluated samples consist of test set samples from the unaffected classes. This is done as error on samples from the deletion set distribution correlates with the unlearning efficacy, and thus removing them leads to a measurement of utility largely independent of unlearning.

In Table 2.3 we show the utilities of the EU-k and CF-k unlearning procedures across all four tests. We observe a negligible impact on utility compared to retraining from scratch, unlike most unlearning

Method	Epochs	Mem (Targeted Error)	Prop. Gen. (Targeted Error)	Test-Error
CIFAR10				
Original	-	3016.0	927.0	16.00
CF-10	6	1453	408	11.07
	14	1305	366	10.63
	30	1226	335	10.24
CF-50	6	758	251	9.42
	14	643	241	9.17
	30	569	229	9.25
Retrain	62	390	184	9.33
CIFAR100				
Original	-	395	70	32.53
CF-10	6	357	56	32.71
	14	348	55	32.60
	30	337	54	32.98
	62	325	57	32.60
CF-50	6	128	47	32.88
	14	141	45	32.12
	30	108	47	32.11
	62	86	36	31.92
Retrain	126	64	31	30.82

Table 2.5: Varying catastrophic forgetting epochs on the IC test. The number of epochs used for fine-tuning can further control the forgetting-efficiency tradeoff without hurting utility.

procedures suggested in existing literature. The only significant difference in error is observed in the I.I.D confusion test, where better unlearning leads to improved utility as the model gets less confused by the mislabelled samples. Note that this is not observed in the IC test as the error is reported on only the unaffected classes, where error is independent of unlearning. Thus, EU-k and CF-k can be used to control the unlearning-efficiency tradeoff at a fixed utility.

In Table 2.4 we show the impact of regularization on utility. We observe that early stopping slightly increases the errors, while cutmix alone reduces them especially in CIFAR100. Given the significant improvement in utility and greater downstream amenability to unlearning, using regularizers like Cutmix seems highly rewarding. Our unlearning procedures do not decrease the utility barring a slight deterioration when the training procedure uses cutmix while the unlearning procedure does not.

2.6 Sensitivity Analysis

We show empirical results on (i) varying the number of unlearning epochs, (ii) layers in the architecture and (iii) samples to be deleted. We also vary the choice of confused classes and ablate the effect of warm restarts in our training procedure. These results demonstrate the robustness of our observations to changes in optimization details, model, selected classes, and training procedure.

2.6.1 Varying the number of unlearning epochs

The original experiments train CF-k models for half the epochs compared to EU-k models. In Table 2.5 we compare the variation of performance among CF-k models at the end of each warm restart while finetuning. While less information is unlearnt on reducing epochs, even six epochs are sufficient for drastic improvements in forgetting, with no significant change in utility (error on full test set). The number of catastrophic forgetting epochs can thus be reduced, and control the forgetting-efficiency tradeoff at constant utility.

2.6.2 Varying the number of layers

In Figure 2.6 we show results of varying k for 3 different ResNet depths: 20, 56 and 110. The IC test is able to detect retained information despite exact unlearning of almost 30% of the final layers. However, on unlearning the final half of the network, its unclear whether most information is removed or the IC test is unable to identify the presence of retained information. CF-k is consistently within a small margin of EU-k demonstrating the catastrophic forgetting is able to lose enough information to match EU while being two times faster.

2.6.3 Varying Amount of Untargeted Removal

In Figures 2.7 and 2.8, we show the forgetting performance when we vary deletion set sizes in tests with untargeted removal: I.I.D Removal and I.I.D Confusion. Here, we use larger sizes than those reported in the main paper as smaller deletion sets show negligible trends in untargeted removal. For detecting effects on property generalization, Error on I.I.D confusion test needs far fewer samples than Error on I.I.D Removal. For memorization, we see that Error is able to distinguish and rank models fairly well whereas MIA works well in the case of I.I.D Removal test but fails completely on the I.I.D Confusion test. CF models continue to be close to EU models here and the gap between them decreases as we add more confusion. Overall, untargeted removal requires much larger deletion sets to show clear forgetting trends as compared to targeted removal, demonstrating the usefulness of strategic sampling.

2.6.4 Varying Amount of Targeted Removal

Now, we study the forgetting performance for partial Class Removal and partial Interclass Confusion. We show results for varying $|S_f|$ from 10% samples of a class to the size of an entire class (as used in the original paper).

First, we present the results of the IC test in Figure 2.9. We see that for memorization all metrics are reflective even when a very small subset of samples is confused. Error and MIA having increasingly better contrast for smaller deletion sets.

Then, we present the results of the Class Removal test in Figure 2.10. The Class Removal test has significantly different behavior when all samples of the class are removed compared to partial class removal. In the case of full Class Removal, all information about the class is removed, and hence an unlearnt model is expected to not classify any sample as the removed class. However, in partial Class Removal, a well generalized model may correctly classify more samples as the affected class, thus leading to the misalignment of utility and forgetting. We observe that MIA seems to have unclear trends in partial Class Removal, sometimes giving a weak signal for unlearning efficacy.

2.6.5 Varying Confused Classes

Throughout our experiments, we only confused the hardest pair of classes in the dataset (Cat and Dog for CIFAR10, Maple Tree and Oak Tree for CIFAR100). In Figure 2.11 we ablate the chosen class pair, grouping the ten classes in CIFAR10 into five pairs to maximize diversity. The five pairs are arranged in increasing order of similarity below along with their bar color:

- Frog (6) - Horse (7): Red
- Bird (2) - Ship (8): Blue
- Airplane (0) - Deer (4): Light Green
- Automobile (1) - Truck (9): Dark Green
- Cat (3) - Dog (5): Black

We can see that the number of confused samples by any model is much higher as we go from left to right, indicating that confusing a similar pair of classes makes unlearning more difficult. Both memorization and property generalization trends across varying levels of unlearning, from Original to Retrain, are consistently preserved. This shows that irrespective of the chosen class pair, the IC test is able to clearly distinguish varying degrees of forgetting.

Sched	CF-10		CF-50	
	MIA	Targeted Error	MIA	Targeted Error
CIFAR10 ($\|S_f\| = 4000$)				
WR	58.66	335	54.24	229
No	58.10	340	53.62	234
CIFAR100 ($\|S_f\| = 400$)				
WR	77.99	58	58.24	41
No	78.87	55	58.57	43

Table 2.6: We compare Warm Restarts and keeping a single learning rate cycle between the same maxLR and minLR. MIA represents memorization while Targeted Error measures property generalization.

2.6.6 Learning Without Restarts

One concern which may arise is whether catastrophic forgetting performs well due to warm restarts in our learning rate schedule. We ablate this effect in Table 2.6 and see that in all cases removing warm restarts has no effect on the degree of catastrophic forgetting.

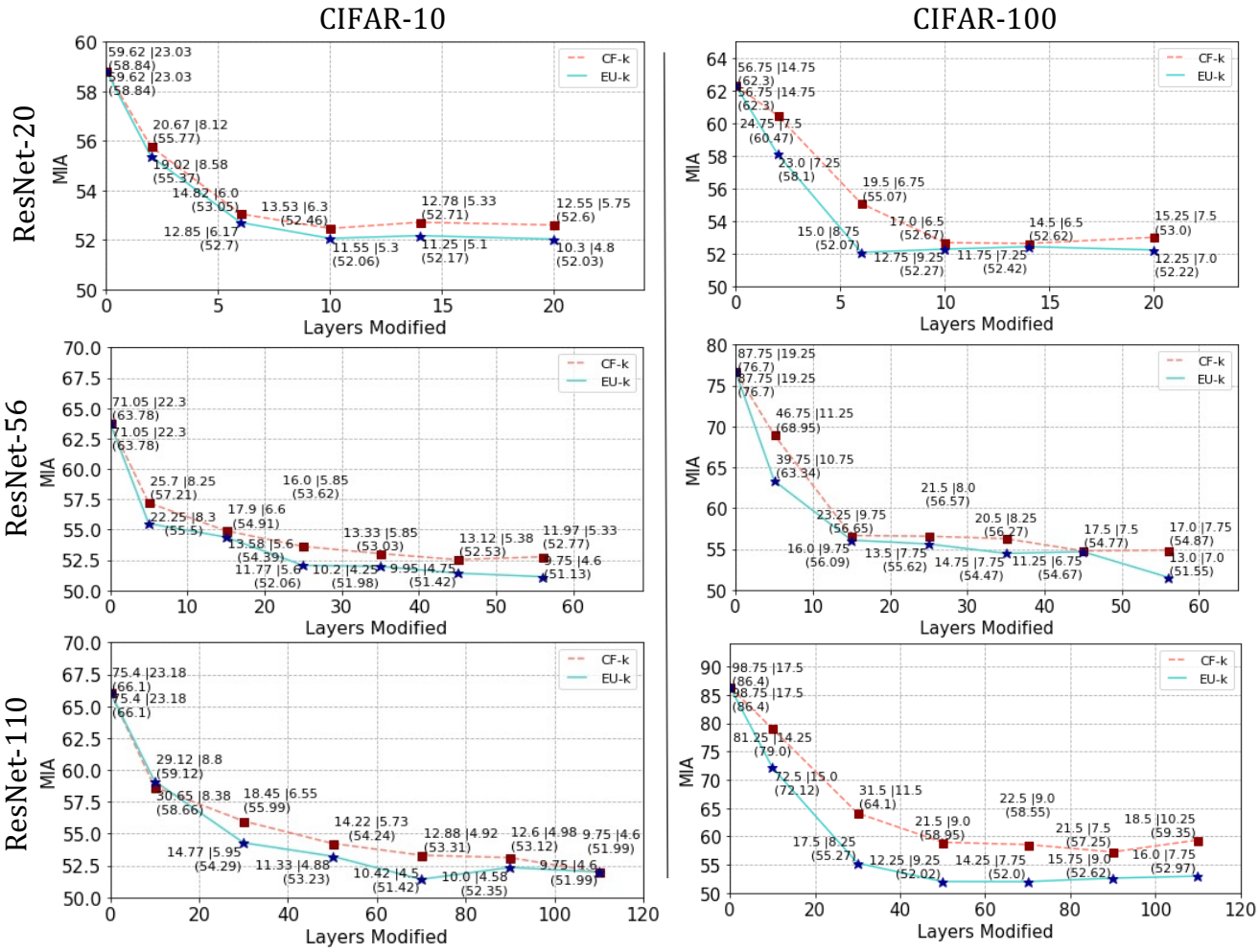

Figure 2.6: We plot the MIA (Y) vs number of layers unlearnt using EU-k (solid blue) and CF-k (dashed red) for different architectures across datasets. For each model (point) we report three forgetting metrics as 'memorization | property generalization (MIA)' with memorization and property generalization computed using targeted error. The leftmost point is the original model while the rightmost EU point is the full retrained model. We observe consistent observations with the main paper across metrics and datasets.

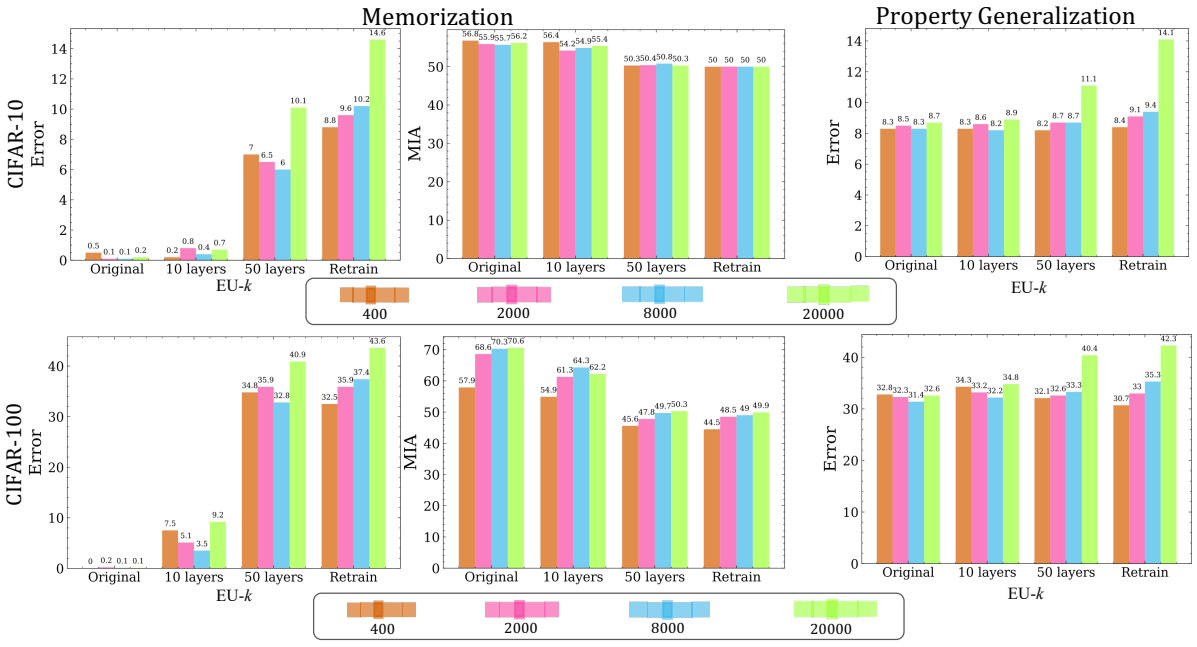

Figure 2.7: Varying $|S_f|$ for I.I.D Removal test. Error seems to distinguish varying levels of memorization, but needs huge deletion sets (50% of dataset size) in the case of property generalization. Moreover, here error has the limitation of misaligning forgetting (↑ is better) and utility (↓ is better).

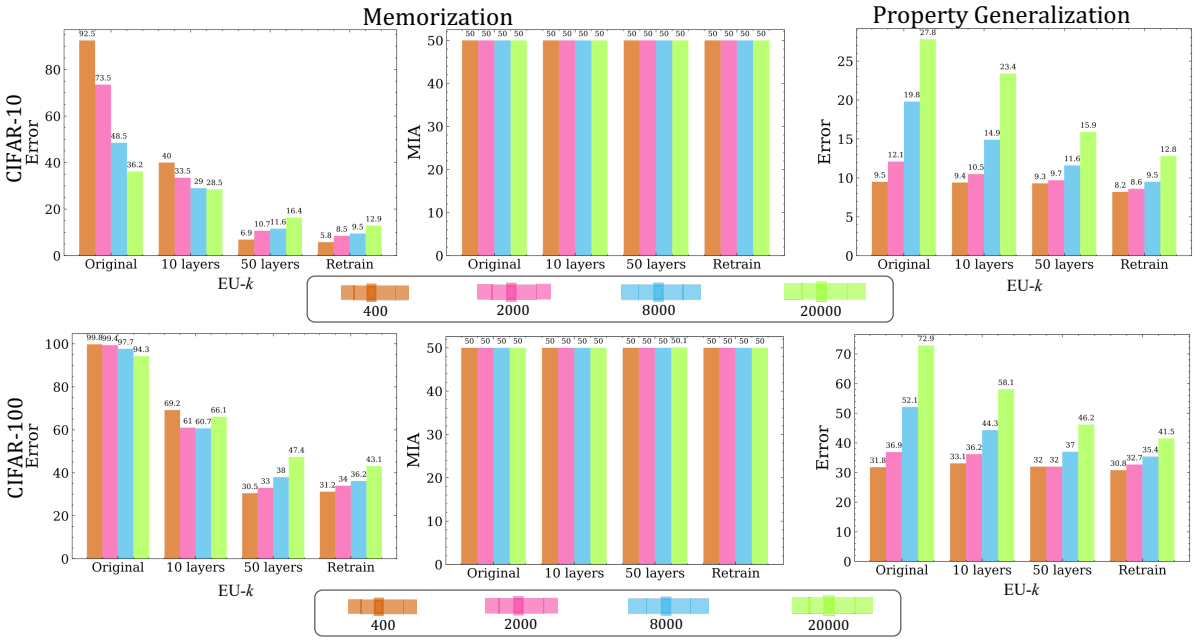

Figure 2.8: Varying $|S_f|$ for I.I.D Confusion test. Error reliably measures memorization even in small deletion sets (1% of deletion set size), though much larger ones (20% of deletion set size) are needed to produce detectable effects on property generalization.

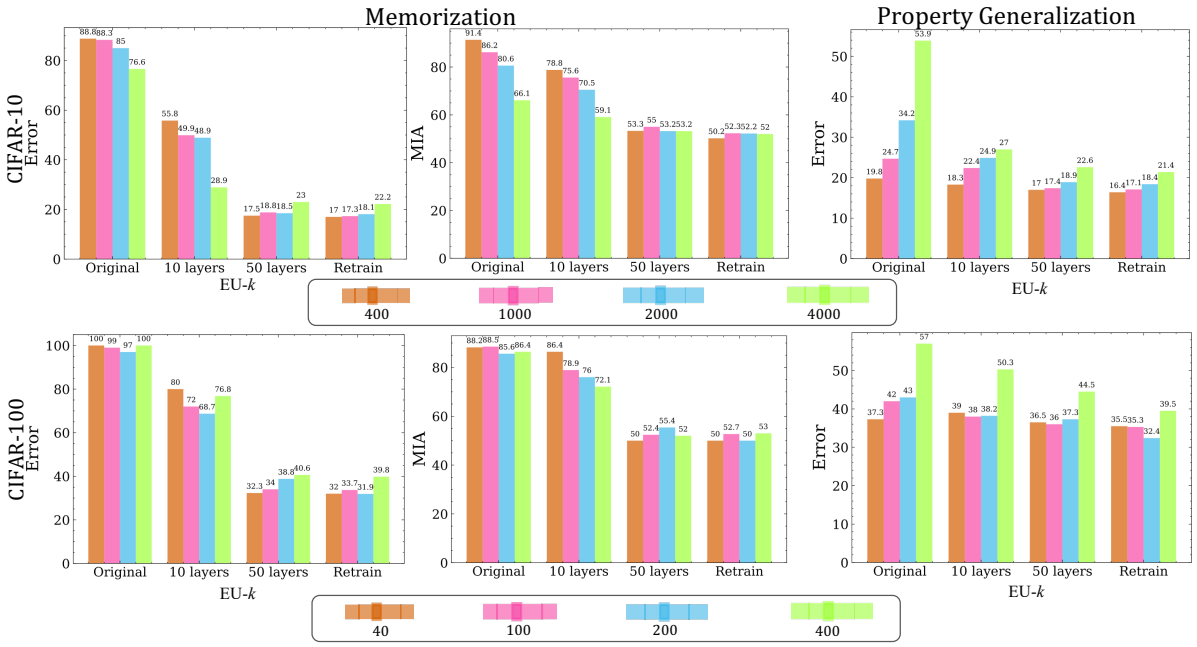

Figure 2.9: Varying $|S_f|$ for IC test. In CIFAR10, at 1% of dataset size, the IC test reliably detects imperfect forgetting across metrics. In CIFAR100, imperfect removal of memorization is detected at 1% of the class size, a noticeable effect on generalization requires a larger deletion set (5% of dataset size).

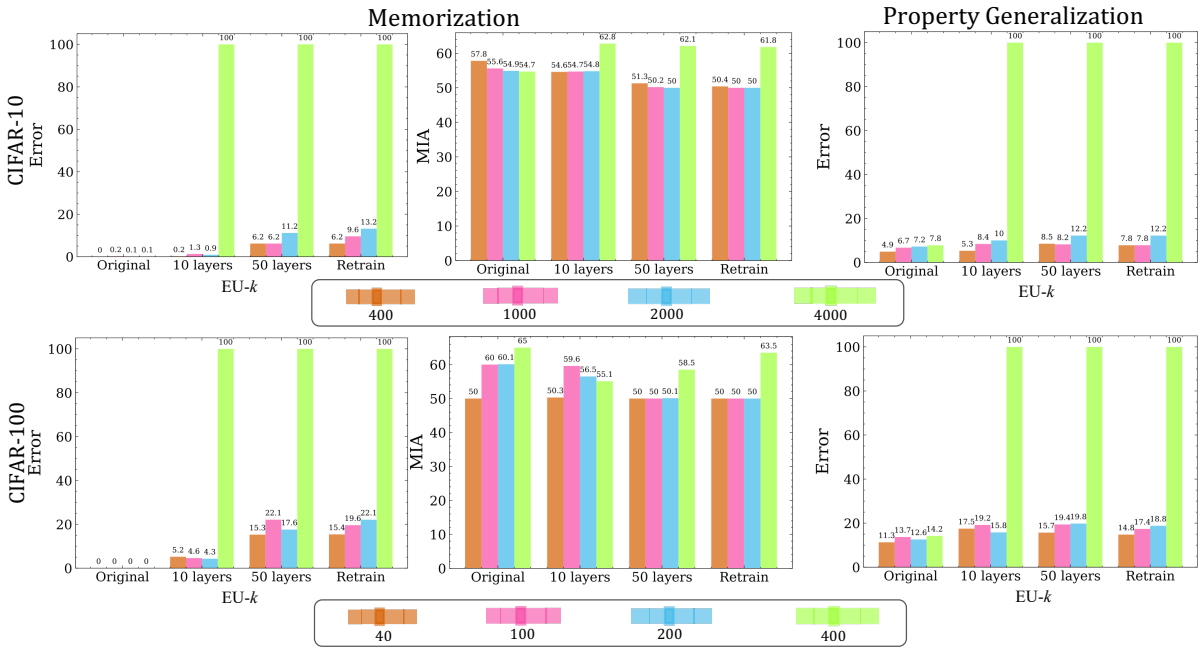

Figure 2.10: Varying $|S_f|$ for Class removal test. The Class removal test is not able to reliably distinguish varying levels of property generalization and provides a weak signal for memorization, particularly for small $|S_f|$.

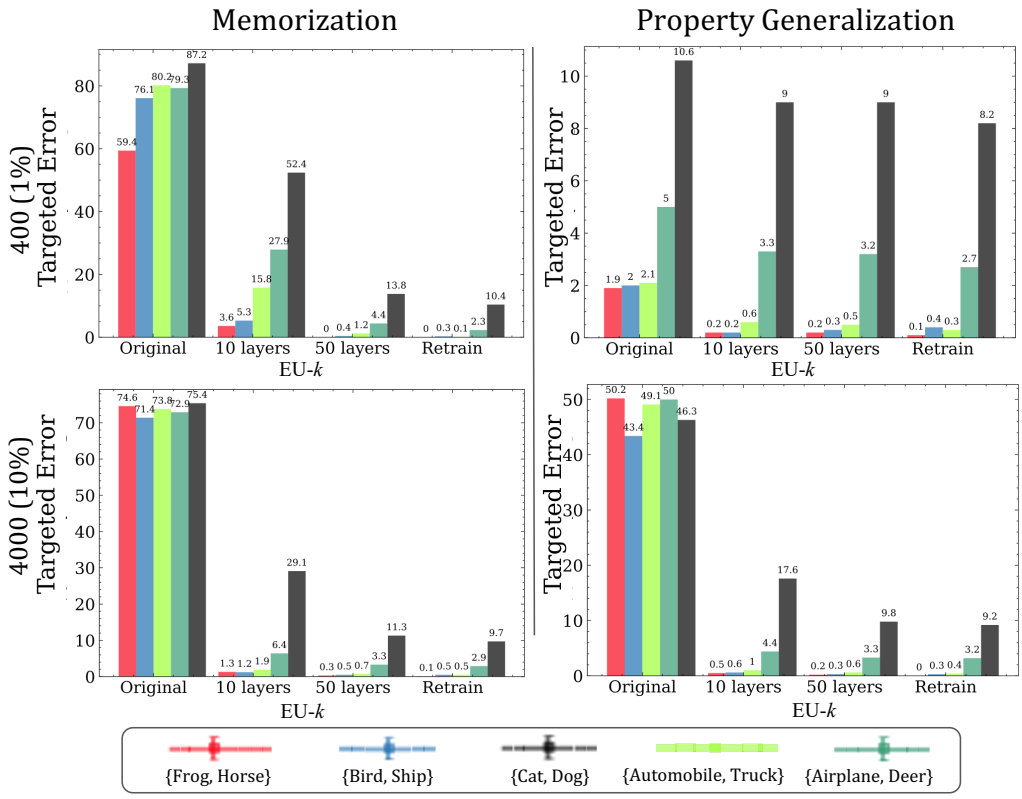

Figure 2.11: Varying confused class pairs on CIFAR10, with the similarity of the classes increasing from left to right in each group of bars. While the IC test reliably detects imperfect forgetting across class pairs, the trends are clearer for more similar classes.

Chapter *3*

Corrective Unlearning[1]

3.1 Introduction

Foundation models are increasingly trained on large and diverse datasets, including millions of web pages and contributions from numerous users and organizations (Schuhmann et al., 2022; Gao et al., 2020). However, data integrity issues significantly impact model performance (Konstantinov and Lampert, 2022; Paleka and Sanyal, 2023) by introducing systemic biases (Prabhu and Birhane, 2021) and adversarial vulnerabilities (Barreno et al., 2006; Sanyal et al., 2021). For instance, a small manipulated subset of web data sources has led to large-scale model poisoning (Carlini et al., 2023), underscoring the vulnerability of these models to such adversarial tactics. Moreover, a critical real-world obstacle is that model developers can often only identify a fraction of the manipulated data, especially when the manipulations are small, imperceptible changes to input or incorrect labels.

Model developers maybe notified of the manipulated data, either through poisoning defenses and other methods for monitoring of the data pipeline (Breck et al., 2019; Wang et al., 2019; Northcutt et al., 2021b) or external information. Due to high costs incurred in training, they may wish to update models trained on the corrupted data, instead of stopping their use. To solve this problem of removing the influence of manipulated data from a trained model, we introduce the concept of *Corrective Machine Unlearning*. This approach aims to efficiently eliminate any detrimental effects from the identified samples, even when the precise nature and extent of the manipulation is unknown. Corrective unlearning has different underlying requirements from the traditional unlearning literature (see Nguyen et al. (2022) for a survey) which is motivated by catering to user data deletion requests in light of privacy regulations (Council of European Union, 2018; California State Leglisature, 2018; Parliament of Canada, 2018). Specifically, corrective unlearning procedures do not need to obtain privacy guarantees on the "unlearned" data. Instead, they *must improve clean-label accuracy on parts of the data domain where model performance*

[1]Goel, S., Prabhu, A., Torr, P., Kumaraguru, P., Sanyal, A. (2024). Corrective Machine Unlearning. Data-centric Machine Learning Research (DMLR) Workshop at The Twelfth International Conference on Learning Representations (ICLR). All figures in this section are taken from the paper.

is adversely affected by the manipulated data while only having access to a representative subset of manipulated samples.

We investigate the application of state-of-the-art unlearning procedures (Kurmanji et al., 2023; Goel et al., 2023; Chundawat et al., 2023a; Foster et al., 2023) to remove adverse effects of two different kinds of manipulations. First, we study a classic poisoning attack (Gu et al., 2019), where a trigger pattern is embedded in a subset of samples, which are then assigned incorrect labels. Such manipulations occur when collecting both features and labels from internet web-pages which adversaries can modify, such as Wikipedia, as demonstrated by Carlini et al. (2023). This can lead to a backdoor where adversaries trigger model misclassifications by inserting the trigger pattern during deployment. Such actions can significantly harm applications, such as autonomous driving (Han et al., 2022). Second, we study the Interclass Confusion test (Goel et al., 2023) where the adversary incorrectly labels samples between two classes thereby entangling the model's representations. Such mislabeling can cause systematic biases in model outputs (Prabhu and Birhane, 2021). Such label-only manipulations can occur when model developers have their own unlabelled datasets but rely on external sources for annotation.

Model developers may eventually recognize compromised data sources and wish to unlearn the influence of this data from previously trained models. We find that many recent unlearning methods, including the traditional gold standard of retraining-from-scratch, fail in the context of corrective unlearning as illustrated in Figure 3.1. Particularly, even knowing 80% of the manipulated data is not enough to remove the adverse effects introduced by manipulating just 1% of the whole training data. However, the Selective Synaptic Dampening (Foster et al., 2023) method is able to remove the effect of BadNet poisoning with just 10% of the manipulated data being identified, showing the tractability of this setting. However, it leads to a significant drop in overall test accuracy, and fails in the Interclass Confusion setting, leaving much to be desired. Overall, this chapter highlights the need for unlearning procedures tailored to removing the influence of manipulated data.

3.2 Ideal Corrective Unlearning

In this section, we formalize the requirements of corrective unlearning, and detail key differences from the traditional privacy-oriented unlearning.

3.2.1 Problem Setting

We initiate our discussion by detailing the ideal corrective unlearning framework, introducing a precise threat model, and identifying specific desiderata.

Scenario: Training sets for large models are often compilations of data from diverse sources such as web pages, platforms like Reddit, data contractors, annotators, user inputs etc. These sources can introduce systematic biases or, more critically, contain data that has been adversarially manipulated, motivating model developers to use corrective unlearning. Crucially, corrective unlearning methods should be able

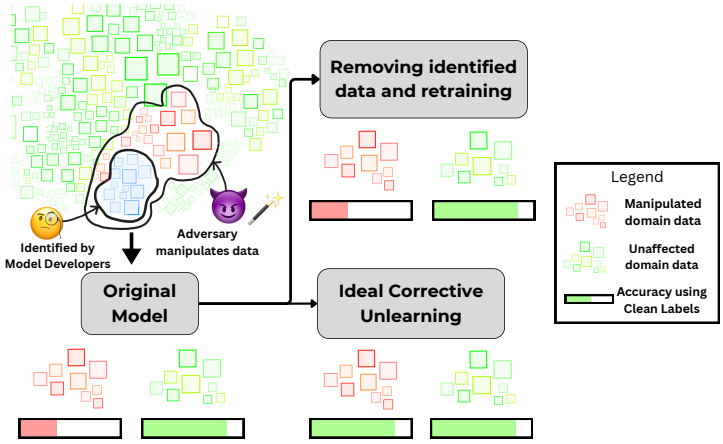

Figure 3.1: Traditionally, retraining after removing identified data is considered a gold standard in unlearning. However, since developers may not identify all the wrong data for unlearning, retraining-from-scratch on remaining data leads to poor clean-label accuracy. Ideally, corrective unlearning procedures should improve accuracy on the affected domain with access to only a representative subset of the wrong data.

to tackle a strong adversarial threat model that allows arbitrary manipulations. In doing so, it's reasonable to expect these unlearning methods can also address problems stemming from naturally occurring benign errors.

Threat Model: Next, we discuss the adversary and model developer's perspective.

Adversary's Perspective: The adversary can arbitrarily manipulate any portion of the input data, including labels in supervised learning scenarios. For example, in poisoning attacks, a trigger is inserted into each manipulated data sample, altering its label to an incorrect one (Han et al., 2022).

Developer's Perspective: Model developers identify some of the compromised data sources after having already trained a model, either through internal monitoring or defenses or external information like tipoffs. While detecting all manipulated data is challenging, it is feasible to be given a small subset which we assume to be representative of the broader set of manipulated data. Since the adversary can apply arbitrary manipulations, the exact manipulation type is unknown to the model developer apriori. The goal of model developers is to remove the adverse effects of the manipulated data from the original model using this small identified representative subset.

Formalization and Notation: Let $\mathcal{X}$ be the data domain, $\mathcal{Y}$ be the label space, and $\mathcal{P}$ be the distribution on $\mathcal{X} \times \mathcal{Y}$. Let $S_{tr} \subset \mathcal{X}$ be the training data, and $S_m \subset S_{tr}$ be the training samples manipulated by the adversary, either by modifying features, the associated training labels, or both. Let $\mathcal{D}_m \subset \mathcal{X}$ be the domain where performance is adversely affected when learning using S_m. For example, in poisoning, $\mathcal{D}_m$ contains samples with the poison trigger. In Interclass Confusion, $\mathcal{D}_m$ consists of samples from the two affected classes. Clearly, $\mathcal{D}_m$ also contains S_m. Finally, let A be the learning algorithm, and $M_o = A(S_{tr})$ be the original trained model.

A corrective unlearning algorithm U_{corr} "improves" the original model (M_o) by removing the influence of S_m. Typically, we expect only a subset of samples to be identified as manipulated, which we denote as the deletion set $S_f \subseteq S_m$. Thus, U_{corr} takes as inputs M_o, S_{tr}, S_f and yields an *unlearned* model M_u. Next, we list the goals of an unlearning procedure.

Desiderata: A corrective unlearning procedure U_{corr} has the following objectives:

(1) *Removing the influence of manipulated samples*: The primary goal is to remove the adverse effect learnt due to the manipulated data S_m. We operationalize this as improving the *clean-label accuracy on* $\mathcal{D}_m$:

$$\mathbb{E}_{(x,y)\sim\mathcal{P}}\left[\mathbb{I}\{h(x) = y\} \mid x \in \mathcal{D}_m\right]$$

where $h = U_{corr}(M_0, S_{tr}, S_f)$. We also compute the clean-label accuracy on the manipulated training set S_m to check if the unlearning procedure "corrects" the manipulation in the training data. It is important to note that while the domain $\mathcal{D}_m$ may be easier to identify for some kind of manipulations like poisoning, it may be more difficult in other cases.

(2) *Maintaining model utility*: Intuitively, the unlearning process should not harm performance on unrelated samples i.e. data outside $\mathcal{D}_m$, retaining model utility. We operationalize this as the overall accuracy ($\mathcal{X} \setminus \mathcal{D}_m$):

$$\mathbb{E}_{(x,y)\sim\mathcal{P}}\left[\mathbb{I}\{h(x) = y\} \mid x \notin \mathcal{D}_m\right]$$

where $h = U_{corr}(M_0, S_{tr}, S_f)$.

This quantity should decrease minimally, and can potentially increase due to a possibly conservative estimate of $\mathcal{D}_m$. For example, the manipulated data may affect the representations learned by the model in unintended ways and thereby impact the utility on unrelated and unexpected parts of the domain.

(3) *Effectiveness with Incomplete Identification*: Corrective unlearning algorithms (U_{corr}) should effectively unlearn adverse effects of manipulations even when the identified subset of the manipulated data S_f is a small representative subset of S_m. This means achieving (1), (2) even when $\frac{|S_f|}{|S_m|}$ is less than one.

(4) *Computation Efficiency*: This is measured as the time taken by the procedure, which should be minimized.

We refer to these desiderata and the associated numbering explicitly throughout the rest of the paper.

3.2.2 Differences from Privacy-Oriented Unlearning

Traditional unlearning seeks to ensure *retrain indistinguishability*: the unlearning procedure U aims to produce a distribution of models that is indistinguishable from one obtained without the forget set. Thus, for some learning algorithm A' which may be different from the original training procedure A, U should produce an indistinguishable distribution of models $U(M_o, S_{tr}, S_f) \sim A'(M_o, S_{tr} \setminus S_f)$. We highlight the distinctive aspects of corrective unlearning as opposed to traditional privacy-focused unlearning, and describe how these differences necessitate changes in unlearning evaluations and method design.

3.2.2.1 No Privacy Requirements

Key Distinction: In the corrective unlearning context, S_f and S_m does not need to be privatized, setting it apart from traditional unlearning.

Implications: Traditional unlearning is designed to meet strict privacy standards, necessitating either : (1) algorithms with theoretical privacy guarantees (Thudi et al., 2022) akin to those provided by differential privacy (Gupta et al., 2021), or at least (2) strong performance against privacy auditing on the data to be forgotten S_f (Golatkar et al., 2020a) such as those performed by Membership Inference Attacks (Shokri et al., 2017). In Chapter 2, we showed rigorous empirical evaluations of the retrain indistinguishability goal are computationally infeasible for deep learning models. Not only is producing a distribution of models expensive, but since A' can differ from the original training procedure, there is a need to search the algorithm space for an A' that produces models indistinguishable from the unlearning procedure. Corrective unlearning bypasses these challenges by setting the practical goal of achieving empirical improvements in model accuracy on samples from the affected domain as the primary success metric (❶).

3.2.2.2 Removal of Incorrect Training Data

Key Distinction: The goal of traditional unlearning is to remove untampered but sensitive user data. However, corrective unlearning removes the influence of samples which were manipulated, either in data, labels or both. This can be particularly challenging for mislabeled data or in multi-class problems, where the corresponding clean version of the data and/or the correct label is unknown.

Implications: Removing accurate samples in traditional unlearning scenarios typically degrades model performance (Golatkar et al., 2020a). Moreover, some unlearning procedures explicitly try to randomize model outputs on forget set samples (Chundawat et al., 2023a; Li and Ghosh, 2023). However, in corrective unlearning, eliminating manipulated samples is expected to significantly enhance model performance on parts of the affected domain $\mathcal{D}_m$ (❶). It may also improve the quality of learned representations leading to increase in overall accuracy (❷).

3.2.2.3 Retrain-from-Scratch is no longer a Gold Standard

Key Distinction: In traditional unlearning, all the data whose influence is to be removed from the model is specified by user deletion requests. However, when identifying manipulated data, it is unrealistic to assume all of it will be found. Thus, in corrective unlearning, $S_{tr} \setminus S_f$ will continue to have manipulated data from $S_m \setminus S_f$ (❸).

Implications: Retraining from scratch on $S_{tr} \setminus S_f$ is the gold standard for traditional unlearning but it is computationally expensive. Therefore, the core challenge for traditional unlearning procedures is achieving computational efficiency (❹). However, in corrective unlearning, as $S_{tr} \setminus S_f$ continues to have manipulated data, unlearning procedures that solely rely on it (Schelter, 2020; Bourtoule et al., 2021; He

Objective	Measurement	Poisoning Figure	IC Test Figure
Removing influence of manipulation on unseen samples (1)	Clean-label accuracy on test set samples from affected domain ($\mathcal{D}_m$)	Figure 3.2	Figure 3.4
Removing wrong predictions on manipulated training samples (1)	Clean-label accuracy on manipulated training samples (S_m)	Figure 3.6	Figure 3.5
Utility (2)	Accuracy on test set samples from unaffected domain ($\mathcal{X} \setminus \mathcal{D}_m$)	Figure 3.3	Figure 3.7

Table 3.1: Summary of figures in terms of quantities reported on the Y-axis, with the X-axis varying $|S_f|$.

et al., 2021; Graves et al., 2021; Goel et al., 2023) perpetuate the adverse effects of the manipulation. This necessitates a methodological inquiry beyond computationally efficient approximations of *retraining from scratch*, which ceases to be a gold standard. This naturally leads to the question *How can we effectively remove the detrimental impacts of S_m using a representative, albeit smaller, subset S_f?*

3.3 Experiments

We study image classification as the broader existing unlearning literature is situated here, only changing the task to corrective unlearning. We benchmark existing unlearning methods in the corrective unlearning setting, across fractions of identified manipulated samples $\frac{|S_f|}{|S_m|}$. We investigate the unlearning of two manipulations: poisoning (Gu et al., 2019) and interclass confusion (Goel et al., 2023).

Roadmap: We report the Experimental Setup in Section 3.3.1. Table 3.1 lists the quantities reported on the Y-axis to measure removal (1) and utility (2). To measure effectiveness at different levels of identification of manipulated samples (3), we vary $|S_f|$ on the X-axis from 10% of $|S_m|$, i.e. a small portion of manipulated samples being used for unlearning, to 100% of $|S_m|$, i.e. all manipulated samples being used for unlearning. Finally, we report computational efficiency (4) of the different methods used in Table 3.3.

3.3.1 Setup Details

Datasets, Models, Manipulation and Deletion Sizes: We use the CIFAR (Krizhevsky et al., 2009) datasets as standard benchmarking datasets in image classification and unlearning. We use the ResNet-9 (Idelbayev, 2018) model for CIFAR10, and WideResNet-28x10 (Zagoruyko and Komodakis, 2016) for CIFAR100. We report results for each dataset for multiple manipulation sizes $n = |S_m|$ as detailed in Table 3.2. In each setting, we vary the deletion set size $|S_f|$ from 10% to 100% of the manipulation size $|S_m|$ at intervals of 10%.

Our standard training procedure A is as follows: We train our models for 4000 steps on CIFAR10, PCAM and 6000 steps on CIFAR100. Each step consists of training on a single batch, and we use

a batch size of 512 throughout. We use an SGD optimizer with momentum 0.9 and weight decay 5e-4, a linear scheduler with t_{mult} = 1.25, and warmup steps as $\frac{1}{100}$ of the total training steps. The same hyperparameters are used during unlearning unless otherwise specified. The setup used for all experiments is a PC with a Intel(R) Xeon(R) E5-2640 2.40 GHz CPU, 128GB RAM and 1 GeForce RTX 2080 GPU.

3.3.2 Unlearning Methods

We benchmark state-of-the-art unlearning methods across paradigms.

(1) Exact Unlearning (EU): This paradigm involves retraining parts of the ML system (Bourtoule et al., 2021; Goel et al., 2023; He et al., 2021) that are influenced by S_f from scratch using $S_{tr} \setminus S_f$.

Method Used: We benchmark the strongest version, retraining the entire model from scratch on $S_{tr} \setminus S_f$ using the original training algorithm A. This is considered an inefficient but gold standard unlearning procedure in prior work.

(2) Catastrophic Forgetting (CF) : Neural Networks suffer from catastrophic-forgetting (French, 1999) - when a model is continually updated without some previously learnt samples, the model loses knowledge about them. Many unlearning methods perform finetuning on $S_{tr} \setminus S_f$ to achieve unlearning of S_f via catastrophic forgetting, and in chapter 2 we show even finetuning just the final layers of the model performs well on the IC test.

Method Used: We use the strongest version of this by using all layers for unlearning. We use the original training procedure A for 1000 steps on $S_{tr} \setminus S_f$.

(3) Modifying learnt parameters with high influence from S_f: This is a training-free class of methods (Golatkar et al., 2020a,b; Peste et al., 2021; Chundawat et al., 2023b) that identifies parameters with information relevant to the forget set using statistics like the Fisher Information Matrix (FIM). It then damages these parameters by adding noise or reducing their magnitude hoping to selectively remove information about S_f.

Method Used: We benchmark the recently proposed Selective Synaptic Dampening (SSD) method which has shown state of the art results in this paradigm (Foster et al., 2023). We extensively tune the weight selection threshold α and weight dampening constant γ. We find that γ should be tuned relative to α for optimal results. For each datapoint, we pick the best result out of runs with $\alpha = [0.1, 1, 10, 50, 100, 500, 1000, 1e4, 1e5, 1e6]$, $\gamma = [0.1\alpha, 0.5\alpha, \alpha, 5\alpha, 10\alpha]$.

(4) Pushing S_f outputs towards random: Some unlearning procedures (Graves et al., 2021; Li and Ghosh, 2023; Chundawat et al., 2023a) push the model towards random outputs on the deletion set.

Method Used: We benchmark Knowledge Distillation from Bad Teacher (BadT) (Chundawat et al., 2023a), a state of the art method in this paradigm, which simultaneously distills from a randomly initialized neural network on S_f, and the original model on the remaining data $S_{tr} \setminus S_f$. We finetune the original model using this procedure for 1000 unlearning steps.

Dataset	#Classes	Model	Poisoning $|S_m|/|S_{tr}|$	IC Test $|S_m|/|S_{tr}|$
CIFAR-10	10	ResNet-9	0.2%, 1%, 2%	1%, 5%, 10%
CIFAR-100	100	WideResNet-28x10	0.2%, 1%, 2%	0.2%, 0.5%, 1%

Table 3.2: Dataset and models along with manipulation sizes for the Poisoning and Interclass Confusion (IC) evaluation.

(5) Alternating between Forget and Retain steps

Method Used: Kurmanji et al. (2023) propose SCRUB and show it performs well on unlearning mislabelled samples when all are identified. The method alternates between forget steps and knowledge preservation steps. The forget step involves doing gradient ascent using the task-loss for S_f. The knowledge preservation step does knowledge distillation from M_o using $S_{tr} \setminus S_f$ as well as optimizing the task-loss on $S_{tr} \setminus S_f$. We finetune the original model using this procedure for 1000 unlearning steps, out of which the forget step is used only in the first 200 unlearning steps as it is recommended in the paper to run it only in the initial iterations. We use a smaller learning rate (0.0025) as the original value leads to stability issues. We tune the hyperparameter α which controls the trade-off between the distillation loss and the task-loss. For each datapoint, we pick the best result out of runs with $\alpha = [0.001, 0.01, 0.05, 0.1, 0.5, 1, 5, 10]$.

Selection of Best Hyperparameters in Unlearning Phase: Most unlearning methods require hyperparameter tuning and this presents a challenge for the model developers on how to pick the best model. Selecting the model with the best validation accuracy may have low removal (❶), especially if the domain affected by the manipulation $\mathcal{D}_m$ is a small fraction of the overall domain $\mathcal{X}$. Moreover, model developers are unaware of the manipulation performed by an adversary, and thus may not be able to precisely isolate the affected domain for validation. In our setting, model developers only have access to S_f; thus even assuming the original training to be incorrect, the correct labels are unknown in multiclass setting. Let the *deletion change* be the fraction of S_f whose prediction by the model differs from the provided label in training. A higher deletion change may indicate more removal. However, note that the deletion change of a trivial model that has no utility (❷) can be quite high. Thus, we propose using **a weighted average of the deletion change and the validation accuracy** to select an unlearnt model that balances removal (❶) and utility (❷). In this work, we weigh them equally.

3.3.3 Unlearning Poisons

Setting: We use the BadNet poisoning attack introduced by Gu et al. (2019) to evaluate the use of unlearning methods to remove backdoors. We manipulate n training images by inserting a trigger pattern that makes 0.3% pixels white at bottom-right positions, re-labeling each of these images to class zero.

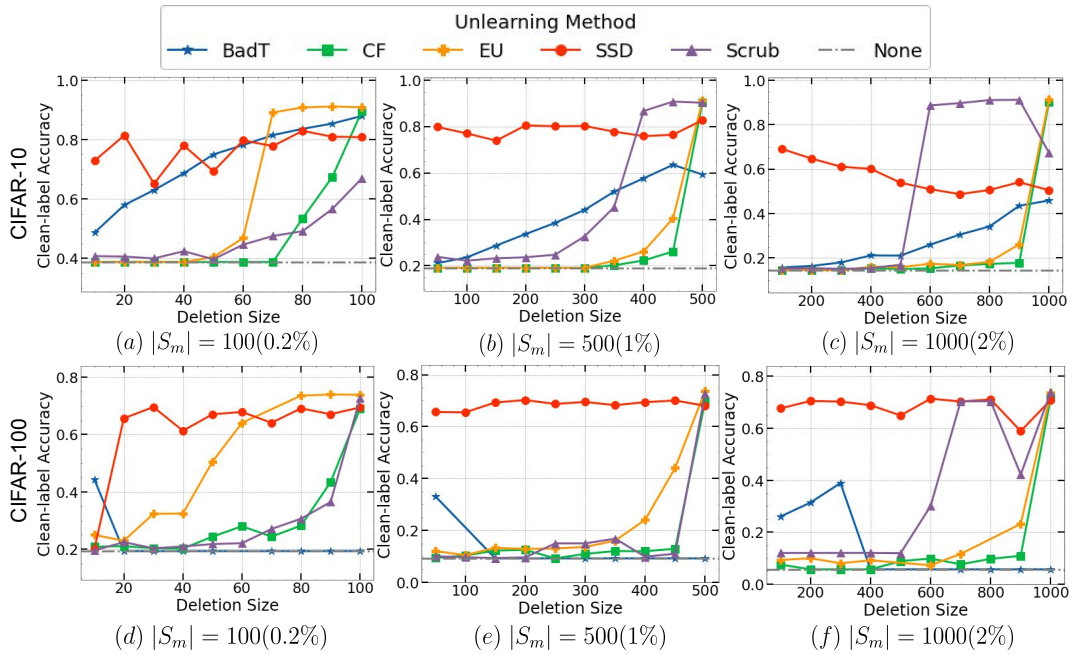

Figure 3.2: Clean-label Accuracy on Test Samples with Poison Trigger. Each method is shown across deletion sizes $|S_f|$ after unlearning ("None" represents the original model). Existing unlearning methods except SSD, including EU which is traditionally considered a gold-standard, perform poorly when $\leq 80\%$ of the poisoned data is identified for unlearning, even when just 1% of training data is poisoned as in (b), (c), (e), (f).

Models trained on datasets containing this manipulation are more likely to label samples containing the trigger pattern as class zero. Here the affected domain $\mathcal{D}_m$ consists of all samples containing the trigger pattern. In this setting, adversaries manipulate both the data features and labels. This can occur when model developers scrape data and corresponding annotations from webpages, such that a subset of these webpages can be manipulated by the adversary.

Results: Figure 3.2 shows clean-label accuracies when the trigger pattern is inserted in all test set samples. EU is the gold standard when all manipulated samples are known, and indeed it achieves the highest accuracy at $|S_f| = |S_m|$. However, it dramatically fails in cases when up to 80% of the manipulated samples are known, even where only 1% (500 samples) of the training data is poisoned (subfigures b, c, e, and f). This shows the insufficiency of the traditional unlearning goal of approximating retraining from scratch on $S_{tr} \setminus S_f$, as the remaining poisoned samples are capable of maintaining their adverse effects, even when their number is small (Gu et al., 2019).

As a consequence, state-of-the-art approaches in unlearning literature like EU, CF, and Scrub perform quite poorly in this setting. BadT shows poor results throughout, as randomizing outputs on S_f conflicts

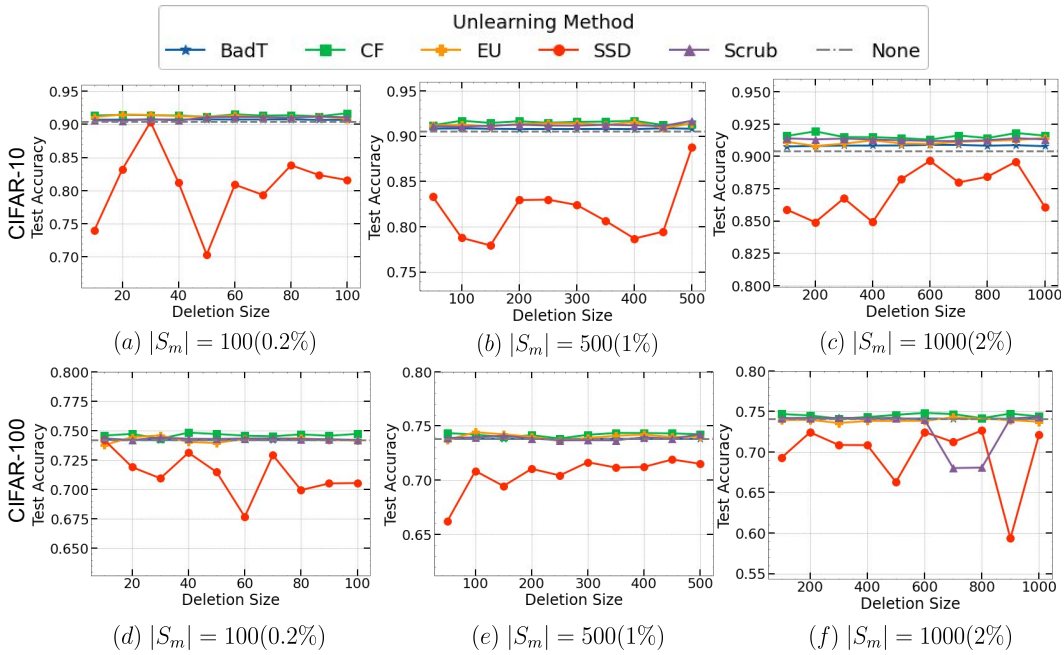

Figure 3.3: Accuracy on Test Samples with No Poison trigger. While other unlearning methods ("None" represents the original model) maintain utility, SSD shows a significant drop across deletion sizes $|S_f|$ across (a)-(f).

with the goal of improving model accuracy on S_f (1). On the contrary, SSD recovers accuracy on $\mathcal{D}_m$ (achieving 1) even with 10% of manipulated samples known, showing the tractability of generalizing removal from a small representative subset of S_m (3). However, as shown in Figure 3.3, SSD leads to significant drops in model utility (2), while other unlearning methods maintain utility throughout. Pruning a small subset of weights is a well-known strategy to mitigate poisons (Wang et al., 2019) as they associate a specific feature with the incorrect label. We believe SSD succeeds in this setting as it can identify weights that learn the BadNet poison effectively even when only a small portion of the manipulation set is known.

Conclusion: Traditional unlearning methods that train on $S_{tr} \setminus S_f$ perform poorly in practical scenarios when all manipulated samples are unknown (3). SSD shows positive results for removing poisons, demonstrating the tractability of corrective unlearning in this setting, though it hurts model utility, leaving scope for improvements. Since SSD works by modifying a small subset of weights, it motivates the usefulness of mechanistic interpretability (Elhage et al., 2021) or influence-function based approaches (Grosse et al., 2023) for removing backdoors at least in small-scale settings.

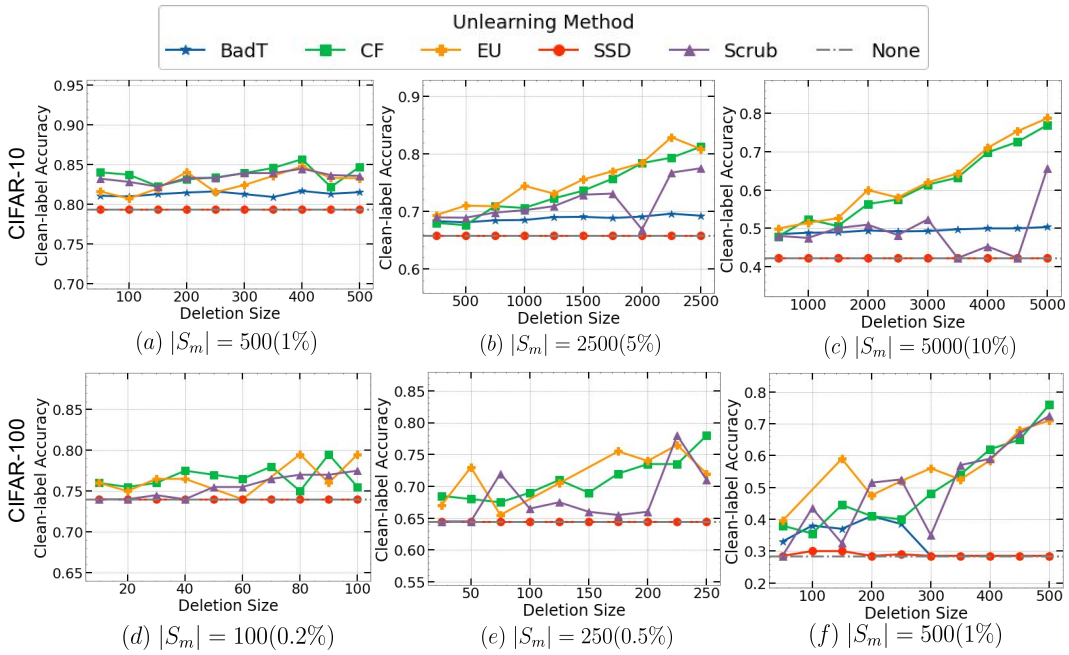

Figure 3.4: Clean-label Accuracy on Test Samples on the Two Confused Classes. We compute clean-label accuracy on the classes A, B used for the Interclass Confusion test, across deletion sizes $|S_f|$. SSD provides no improvements over the original model (represented as "None"), and other unlearning methods also require a large fraction of the manipulated data to be identified for unlearning. In the lower manipulation size setting (a) and (d), the model outputs on unseen samples are not affected much, so we show unlearning trends on manipulated train samples below.

3.3.4 Unlearning Interclass Confusion

Setting: We use the Interclass Confusion (IC) test as a strong evaluation for the use of unlearning methods to remove the influence of mislabels. In the IC test, two classes A and B are picked, and $\frac{n}{2}$ samples from both classes are selected, and their label is changed to the other class. Models trained on datasets containing this manipulation are more likely to confuse these classes, i.e. predict A samples as B and vice-versa. The affected domain $\mathcal{D}_m$ consists of all samples from class A and class B. For CIFAR10, we confuse the Cat and Dog classes, and for CIFAR100 maple and oak tree, which is consistent with the setup in Chapter 2.

The IC test applies in the setting where the adversary can only manipulate labels, such as when model developers outsource annotations for their own data. Mislabels between two classes can also occur due to systematic biases in the labelling process, or misinterpretation in annotation guidelines on how to distinguish the classes. Manipulating only labels may appear to be a weaker setting compared to poisoning. However, unlike poisoning where a small subset of weights may be associated with the trigger

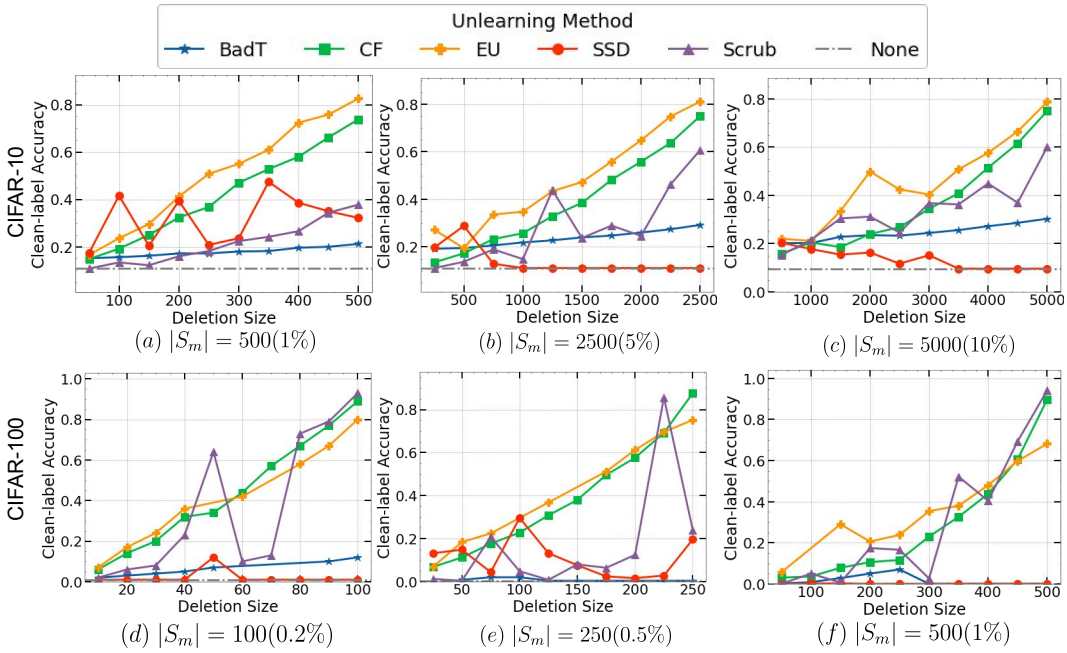

Figure 3.5: Clean-label Accuracy on Manipulated Training Samples S_m with Interclass Confusion for different unlearning methods ("None" represents the original model) across deletion sizes $|S_f|$. Existing unlearning methods perform poorly when $\frac{|S_f|}{|S_m|}$ is lower. Even the smallest setting (a, d) shows clear unlearning trends.

and can be targeted for unlearning, the IC test can have a more uniform effect across weights, confusing the learnt representations of clean samples without any specific triggers. We hypothesize unlearning procedures like SSD that modify specific parameters may be less effective for such settings.

Results: In Figure 3.4, we see that EU, CF, and Scrub show gradual improvement in removal (1) as larger fractions of the manipulated set are identified. BadT performs poorly across deletion set sizes, similar to poisoning. While SSD, a mechanistic intervention that prunes certain weights, showed promising results for poison removal, it completely fails at removing interclass confusion. Finally, while the smallest manipulation size (subfigures a, d) for Interclass Confusion did not show significant effects on unseen samples from class A, B, Figure 3.5 shows unlearning methods continue to give wrong predictions on the class A, B samples used for training. This emphasises the need to check unlearnt model outputs on unseen training samples from the affected domain $\mathcal{D}_m$ in addition to test samples from $\mathcal{D}_m$.

Conclusion: The failure of SSD in this setting highlights the need for evaluating diverse manipulations to test corrective unlearning procedures. Traditional unlearning procedures have poor removal (1) when small subsets of the manipulation set are identified (3). Overall, there is scope for designing better corrective unlearning methods that achieve desiderata 1-3 across different manipulation types.

3.4 Related Work

Learning from manipulated data: The adverse effects of manipulated training data on machine learning models are well-documented across objectives like fairness (Konstantinov and Lampert, 2022), robustness (Sanyal et al., 2021; Paleka and Sanyal, 2023), and adversarial reliability (Tian et al., 2022). One line of defense is designing training strategies more robust to these issues, see Song et al. (2022) for a survey on learning with mislabels. However, learning robust models from manipulated data is a hard problem as reduced sensitivity to such minority data populations can harm accuracy and fairness (Feldman and Zhang, 2020; Sanyal et al., 2022). Unlearning specific samples which are discovered to be manipulated can be a complementary mitigation approach. Further, we hope corrective unlearning procedures are compared using the same original model, to ensure improvements are due to the unlearning procedure rather than properties of the original training procedure or model.

How to detect manipulated data? A prerequisite to the corrective unlearning task is detecting a representative subset of manipulated data. Fortunately, this has long been studied (Brodley and Friedl, 1999), with prior work detailing techniques to discover mislabeled (Pleiss et al., 2020; Northcutt et al., 2021a), biased (Prabhu and Birhane, 2021; Jiang and Nachum, 2020) and poisoned (Chen et al., 2019; Wang et al., 2019) data. Further, compromised data sources can be identified using web security and data collection practices. We assume the model developers employ such strategies for monitoring their data sources. However, they cannot simply throw away the trained model when manipulated data is found due

to expensive retraining costs. We study how to cheaply mitigate adverse effects on such models using unlearning.

Known Manipulations or Correct Labels: If the type of manipulation is known, one may employ manipulation-specific mitigation techniques such as poisoning (sometimes referred to as trojan) defences (see Goldblum et al. (2022) for a survey). We restrict the scope of our work to not knowing the precise manipulation, and study the use of unlearning as a broader panacea procedure across unknown data manipulations. Finally, if the samples can be corrected through re-annotation, one may also use knowledge editing techniques (Bau et al., 2020; Mitchell et al., 2022).

Unlearning: Prior work in designing unlearning procedures is motivated by privacy applications, and aims to achieve *retrain indistinguishability* (Ginart et al., 2019; Golatkar et al., 2020a), that is to create a distribution of unlearnt models indistinguishable from retraining from scratch without the data to be deleted. In Section 3.2.2 we discuss differences in corrective unlearning desiderata from retrain indistinguishability. "Exact Unlearning" procedures ensure the unlearnt model never sees the data whose influence is to be deleted by design of the training procedure (Bourtoule et al., 2021; Schelter, 2020). The empirical results of EU in Section 3.3 show how these approaches may not suffice for corrective unlearning when the full manipulation set is unknown. Moreover, such methods drastically deteriorate in efficiency as the as the number of samples to delete increase (Warnecke et al., 2021). This has led to "Inexact Unlearning" proposals, and we use state of art methods in image classification from different paradigms for our experiments:

- Modifying parameters which influence forget set outputs (Golatkar et al., 2020a; Peste et al., 2021; Ma et al., 2023) - We benchmark Selective Synaptic Dampening (SSD) (Foster et al., 2023).
- Randomizing model outputs on the data to be deleted (Graves et al., 2021; Chundawat et al., 2023b; Tarun et al., 2023) - We benchmark Knowledge Distillation from Bad Teacher (BadT) (Chundawat et al., 2023a).
- Finetuning based approaches only using retained samples (Warnecke et al., 2021; Yao et al., 2023b; Jang et al., 2023; Eldan and Russinovich, 2023; Chen and Yang, 2023) - We benchmark Catastrophic Forgetting (CF), as Goel et al. (2023) show it works well on the Interclass Confusion test.
- Alternating between Forgetting and Preservation Steps - We use SCRUB as Kurmanji et al. (2023) show it works well on the Interclass Confusion test.

A group of works (Izzo et al., 2021; Wu et al., 2020; Gupta et al., 2021; Neel et al., 2021; Thudi et al., 2022; Sekhari et al., 2021) also study unlearning procedures on convex or linear models with theoretical guarantees inspired from differential privacy (Dwork et al., 2006), but in this work we focus on deep models. Finally, Goel et al. (2023); Kurmanji et al. (2023); Sommer et al. (2022) consider unlearning of mislabelled or poisoned samples, but only as a stronger evaluation for the privacy-oriented objective of retrain indistinguishability. We show retraining cannot be used as a gold standard for corrective

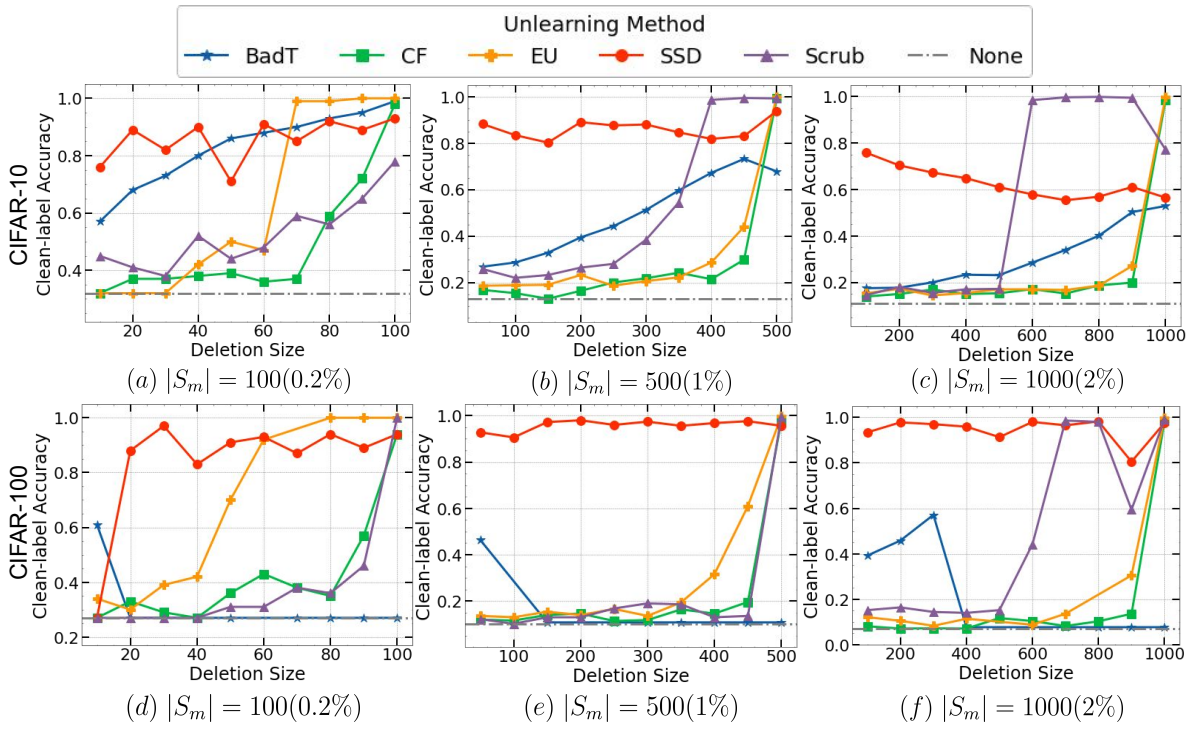

Figure 3.6: Clean-label Accuracy on Manipulated Train Samples S_m with Poison Trigger. Each method is shown across deletion sizes $|S_f|$ after training with adversarial poisoning ("None" represents the original model). Trends mimic results for clean-label accuracy on unseen samples with the poison trigger.

unlearning when only a subset of manipulated samples is identified (3), which leads to the insufficiency of unlearning methods geared towards indistinguishability from retraining for corrective unlearning.

3.5 More Results for Completeness

To ensure completeness, we now provide results that are less insightful, and thus not included in the above sections.

3.5.1 Clean-label Accuracy on Manipulated Training Samples after Unlearning of Poisons

To measure the removal of mislabelling on poisoned training samples, we report clean-label accuracy on S_m in Figure 3.6. The trends across unlearning methods are similar to the ones on unseen samples from the affected domain $\mathcal{D}_m$ reported in Figure 3.2, though the absolute accuracies after unlearning are higher as expected from training samples in comparison to test set samples.

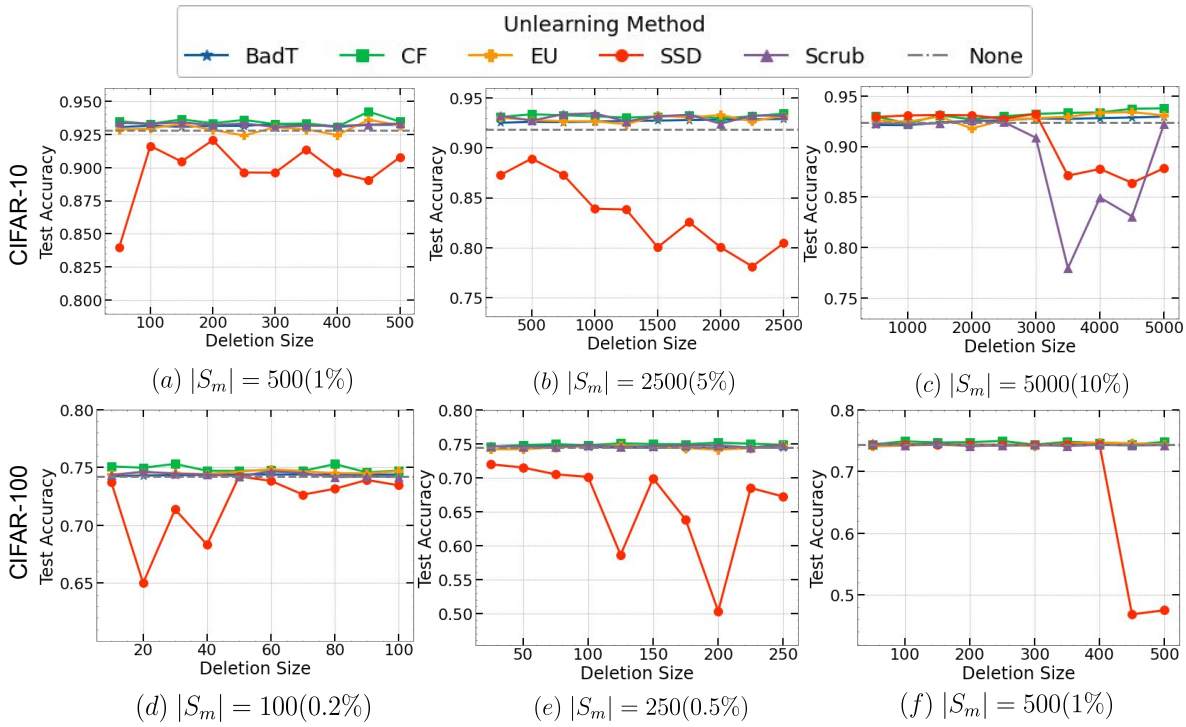

Figure 3.7: Accuracy on Test Samples from classes other than the two confused. Except SSD which shows drops in utility, we see similar accuracies across different unlearning methods across deletion sizes $|S_f|$ after training with Interclass Confusion ("None" represents the original model).

3.5.2 Utility after Unlearning of Interclass Confusion

We report accuracies on unseen samples from the classes not manipulated by interclass confusion. These samples can be considered to belong to the same distribution as $S_{tr} \setminus S_m$. In Figure 3.7 we plot the utilities across deletion set sizes for IC test. We find methods maintain accuracy, and EU, CF even show minor (0.5-1%) gains when most of the manipulated data is known. This is not surprising as removing the effect of manipulations can improve learnt representations and the overall utility of the model.

3.5.3 Computational Efficiency of Unlearning Methods

In Table 3.3 we report average unlearning times of different unlearning methods. In the case of EU and CF, while more efficient relaxations have been proposed (Goel et al., 2023; He et al., 2021; Graves et al., 2021), we retrain from scratch to perform the strongest unlearning, which we still find to be insufficient.

Method	**Time** (minutes)
EU	49.93
CF	10.52
Scrub	16.86
SSD	1.80
BadT	33.19

Table 3.3: Unlearning Time by Method

Chapter *4*

Unlearning Dual Use Knowledge from LLMs[1]

4.1 Introduction

Until now, we have seen the application of unlearning in purely a supervised image classification setting. In this chapter, we explore the use of unlearning for Large Language Models (LLMs). Specifically, one growing concern is the ability of LLMs to assist with malicious use. In particular, LLMs may aid actors in planning bioattacks (Sandbrink, 2023) and procuring pathogens (Gopal et al., 2023). Moreover, LLMs can assist users in synthesizing dangerous chemicals (Boiko et al., 2023) or conducting cyberattacks (Bhatt et al., 2023). In response to these emergent hazardous capabilities, major AI labs have developed frameworks to measure and mitigate biological, cybersecurity, and chemical hazards posed by their models (Anthropic, 2023; OpenAI, 2023b, 2024). Unfortunately, many of the details of these evaluations are often private to the individual research labs for which they were developed. We use WMDP, an open-source evaluation procured by Li et al. (2024) to measure the unlearning of hazardous biosecurity and cybersecurity knowledge from LLMs.

We examine an autoregressive language model designed to process prompts like *How can I synthesize anthrax?*" and generate completions like *To synthesize anthrax, you need...*"). Our objective is to diminish the model's proficiency in responding to queries concerning dangerous information (e.g., synthesizing anthrax) while preserving its capability to respond to inquiries about non-dangerous information (e.g., culturing yeast). We define this objective as reducing a model's Question-Answer (QA) accuracy on WMDP while upholding its performance on general competence benchmarks such as MMLU and MT-Bench.

Unlike unlearning for copyright or privacy concerns, we do not assume access to questions from WMDP. This is because our focus lies in unlearning methods capable of generalization: unlearning hazardous knowledge with access to a representative set of samples from a similar distribution. We introduce CUT, an unlearning method that removes hazardous knowledge without significantly compromising

[1]Li, N., Pan, A., Gopal, A., Yue, S., Berrios, D., Gatti, A.,Li, J.D., Dombrowski, A.K., Goel, S., et al. (2024). The WMDP Benchmark: Measuring and Reducing Malicious Use With Unlearning. arXiv preprint arXiv:2403.03218. All figures in this section are taken from the paper.

general model performance. CUT capitalizes on the concept that model representations encapsulate worldly knowledge and that these representations can be *controlled* to steer the model (Ilharco et al., 2023; Zou et al., 2023a; Turner et al., 2023). Essentially, CUT guides the model to possess a novice-level grasp of hazardous knowledge. We formulate a loss function comprising a forget loss and a retain loss. The forget loss nudges the model representations towards those of a novice, while the retain loss constrains the extent of general capabilities removal.

We adapt the SSD (Foster et al., 2023) and SCRUB (Kurmanji et al., 2023) unlearning methods from supervised image classification to language modelling, showing they perform much worse than CUT. CUT also drastically outperforms LLMU, a recent unlearning method proposed for LLMs. We do find that CUT reduces performance on related but less hazardous fields like introductory virology and computer security, while also slightly reducing model fluency. We believe this CUT is just a demonstration that paves the way for better unlearning methods for removing dual-use domains of knowledge from LLMs.

4.2 Methodology

In this section, we describe the unlearning method proposed in Li et al. (2024), and the evaluations and baselines needed to understand the results that follow.

4.2.1 Method - Contrastive Unlearn Tuning (CUT)

CUT is inspired from Representation Engineering (Zou et al., 2023a), where activations are steered towards a novice using the forget loss, while the retain loss preserves other knowledge.

Forget loss: To determine which specific knowledge areas to eliminate (for example, in cybersecurity), CUT identifies these areas using specific terms (such as exploit development or penetration testing).

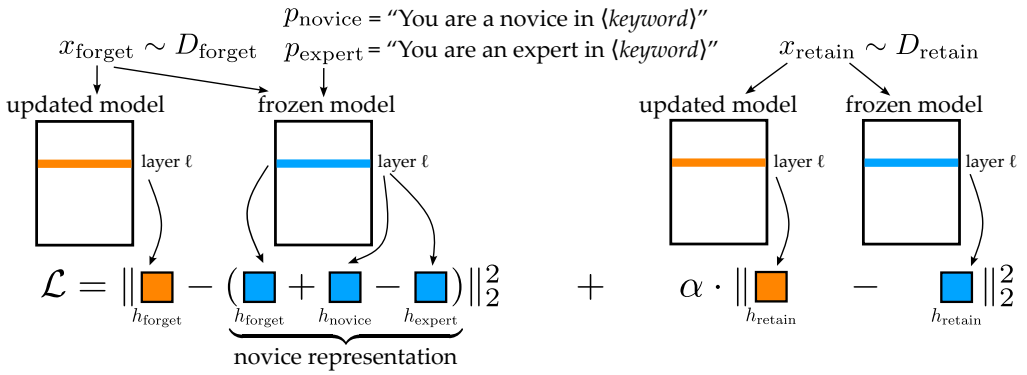

Figure 4.1: CUT optimizes a contrastive loss: a forget component that steers model activations on hazardous data (x_{forget}) towards a novice, and a retain component, which preserves activations on other data (x_{retain}). A multiplicative factor c omitted in the figure for simplicity is used to control this tradeoff.

Eliminating knowledge associated with these terms allows us to effectively remove dangerous knowledge. To influence model behavior, CUT employs control vectors as outlined in the studies by Zou et al. (2023) and Ilharco et al. (2023). For any given term, like "penetration testing," a vector for unlearning control represents in the model's activation domain the lack of knowledge associated with that term. Utilizing these unlearning control vectors for model adjustment offers greater precision than moving towards a vague direction.

For calculating these vectors for unlearning control, CUT considers having access to $M_{\text{updated}}(\cdot)$, which are the hidden states at a certain layer ℓ after unlearning, and $M_{\text{frozen}}(\cdot)$, the hidden states from the original, unchanged model at the same layer ℓ. For a given $\langle$keyword$\rangle$, CUT sets $p_{\text{novice}} =$ "You are a novice at $\langle$keyword$\rangle$" and $p_{\text{expert}} =$ "You are an expert at $\langle$keyword$\rangle$", then compute

$$h_{\text{control}}(\langle\text{keyword}\rangle) = M_{\text{frozen}}(p_{\text{novice}}) - M_{\text{frozen}}(p_{\text{expert}}).$$

To modulate activations, CUT uses a designated forget dataset D_{forget} and define:

$$\mathcal{L}_{\text{forget}} = \mathbb{E}_{x_f \sim D_{\text{forget}}} \left\| \mathrm{M}_{\text{updated}}(\mathrm{x_f}) - (\mathrm{M}_{\text{frozen}}(\mathrm{x_f}) + \mathrm{c} \cdot \mathrm{h}_{\text{control}}) \right\|_2^2,$$

where h_{control} represents a unlearning control vector selected randomly from our collection.

Preservation loss: To minimize the loss of general abilities through unlearning, CUT applies an ℓ_2 penalty to push the model's activations towards those of the original, unchanged model. Using a retain dataset D_{retain}, the preserve loss is calculated as:

$$\mathcal{L}_{\text{retain}} = \mathbb{E}_{x_r \sim D_{\text{retain}}} \left\| \mathrm{M}_{\text{updated}}(\mathrm{x_r}) - \mathrm{M}_{\text{frozen}}(\mathrm{x_r}) \right\|_2^2.$$

Full loss: The total loss, depicted in Figure 4.2.1, integrates the forget and retain losses through a weighted sum:

$$\mathcal{L} = \mathcal{L}_{\text{forget}} + \alpha \cdot \mathcal{L}_{\text{retain}}.$$

CUT fine-tunes the model's parameters to reduce this combined loss. To forget various knowledge distributions, CUT alternates between gradient updates (for example, first on the biosecurity knowledge, then on cybersecurity, and so forth). Adjusting only a few layers is enough, which conserves memory and facilitates efficient unlearning in large models (with 34 billion parameters).

Forget and retain datasets. D_{forget} for biosecurity is collected from PubMed papers. For cybersecurity, it is collected by crawling GitHub for relevant documents. Li et al. (2024) set D_{retain} to be Wikitext (Merity et al., 2016).

4.2.2 Evaluations

In this section we discuss the QA evaluation to measure unlearning performance, evaluations to ensue the model retains general capabilities, and an evaluation that probes whether the information is scrubbed from model internals as well.

4.2.2.1 QA Evaluation

Li et al. (2024) introduce the **W**eapons of **M**ass **D**estruction **P**roxy (WMDP) benchmark, a dataset of expert-written, multiple-choice questions in biosecurity (WMDP-Bio), and cybersecurity (WMDP-Cyber). The dataset measures hazardous dual-use knowledge in these domains, so the unlearning goal is to reduce question-answer (QA) accuracy on WMDP.

4.2.2.2 Retaining Capabilities

It is imperative that the unlearnt model maintains as much of the original performance on capabilities unrelated to the dual-use knowledge removed. We measure this using the MMLU benchmark (Hendrycks et al., 2020). Further, unlearning biosecurity knowledge is most likely to have ripple effects on biology knowledge, and to measure this we report MMLU performance on topics similar to biosecurity (college biology, virology). Similarly, for cybersecurity, we report performance on the college computer science and computer security section of MMLU. Finally, the LM outputs should not become less fluent, and we evaluate this using MT-Bench (Zheng et al., 2023), a multi-turn conversation and instruction-following benchmark.

4.2.2.3 Probing Evaluation

While evaluating QA accuracy measures an API-access threat model where only outputs or logits may be available, the model internals may still continue to retain the knowledge. One example is if the unlearned model refuses to answer all queries, while still containing hazardous knowledge. To test against this failure mode, we train linear probes on all internal layers to see if the information to be unlearned still exists.

4.2.3 Baselines

Here, we describe the baselines we compare CUT with. While these baselines have been proposed for different tasks, we make our best attempt to adapt them to our setting.

4.2.3.1 LLMU

Yao et al. (2023b) propose Large Language Model Unlearning (LLMU) which uses a mix of gradient ascent to increase the forget set loss, and finetuning towards random data. We apply a grid search over the learning rates $[1 \times 10^{-4}, 5 \times 10^{-4}, 1 \times 10^{-3}, 5 \times 10^{-3}]$, the number of steps $[500, 750, 1000]$, and the forget weight $[0.5, 1, 2]$.

4.2.3.2 SCRUB

SCRUB (Kurmanji et al., 2023) was discussed earlier in the context of image classification, for which it is proposed. Here we adapt SCRUB for LLM unlearning. To do this, we cycle between forget data and retain data epochs, maximizing KL divergence of logits between the student and teacher model on the forget set, and minimizing it on the retain set. The retain set epochs also includes a task-specific loss with gold labels to maintain performance. We use the same forget set and retain sets as the CUT experiments, and with log perplexity on Wikitext as the task-specific loss. We tune the α hyperparameter at values $[1 \times 10^{-4}, 1 \times 10^{-3}, 1 \times 10^{-2}, 1 \times 10^{-1}, 1, 10]$, to search over loss weightings between knowledge distillation and the task-specific loss. We do this as a grid search with learning rates being $[1 \times 10^{-5}, 5 \times 10^{-6}, 2 \times 10^{-6}]$. We use 600 unlearning steps in total, doing the forget step only for 300 as it is recommended in Kurmanji et al. (2023) to stop it earlier. In the high learning rate case, i.e. $lr = 1e-5$ we also try doing only 400 unlearning steps in total, with only 100 forget steps. Each sample of our dataset is truncated to 200 characters, and we use a batch size of 2. As shown in Chapter 3, SCRUB performs poorly when most training samples relevant to removal are not available. This could be one of the reasons why SCRUB performs poorly in our setting.

4.2.3.3 SSD

Selective Synaptic Dampening (SSD) (Foster et al., 2023) belongs to a class of methods which find parameters in the model that are differentially more important for the forget set than the retain set. While the method was originally developed for image classification, we adapt it for autoregressive language modeling by altering the loss function to log-perplexity on the forget set and retain set. We grid-search on the threshold $[0.1, 0.25, 0.5, 1, 2.5, 5]$ and constant for dampening $[1 \times 10^{-5}, 1 \times 10^{-4}, 1 \times 10^{-3}, 1 \times 10^{-2}, 1 \times 10^{-1}, 1]$, the two main hyperparameters for SSD. We converged on these ranges after initial manual hyperparameter exploration for our task and datasets.

4.3 Experiments

4.3.1 Setup Details

Models: We use ZEPHYR-7B-BETA (Tunstall et al., 2023) as it is among the best open-source LMs at 7 billion parameters, and similarly YI-34B-CHAT (01-ai, 2023) at 34 billion parameters. The performance of GPT-4 (OpenAI, 2023a) is reported as an upper bound.

Output Setup: We use a zero-shot question-answer format, taking the top logit between `A`, `B`, `C`, and `D` as the answer choice. For comparison, we also benchmark GPT-4 zero-shot on each of these tasks. As language models are sensitive to the prompting scheme (Sclar et al., 2023), we use `lm-evaluation-harness` (Gao et al., 2021) to standardize prompts. The same setup is used for MMLU, and the standard setup is used for MT-Bench.

Probing Setup: We train a 4-way (Options A, B, C, D) linear classifier on the unlearned CUT models using half of WMDP-Bio and WMDP-Cyber. The other half is held out for evaluation of accuracies, where a high accuracy indicates an unlearning failure, i.e. the layer representation still contains WMDP-relevant knowledge. We test this across all layers of each model.

4.3.2 Results

4.3.2.1 Output Results

As shown in Table 4.1, for both ZEPHYR-7B and YI-34B, CUT is able to drop performance to near random accuracy on WMDP-Bio and WMDP-Cyber, while other baselines struggle to drop accuracy on WMDP-Bio and WMDP-Cyber without crippling model performance on MMLU. Figure 4.3 shows that CUT is pareto-optimal over the baselines across the unlearning-utility tradeoff. Furthermore, Figure 4.4 illustrates that CUT maintains its effectiveness in areas associated with biology (college biology) and computer science (college CS) within the MMLU topics, indicating a higher degree of unlearning accuracy compared to the baseline methods. Nonetheless, there is a significant decline in performance on topics closely related to biosecurity (virology) and cybersecurity (computer security) when using CUT. This points to an opportunity for future research to enhance the preservation of related skills while undergoing the unlearning process. This may be because of using a generic retain set Wikitext, which doesn't provide fine-grained feedback on where to draw the line between what knowledge to unlearn and retain. Finally, CUT manages to sustain its performance on MT-Bench, with only a slight decrease of 0.13 on ZEPHYR-7B and 0.54 points on YI-34B (out of a total of 9). However, as CUT still experiences some decline on MT-Bench, especially with YI-34B, there's a necessity for advanced unlearning techniques that maintain capabilities.

Model	Method	WMDP (↓)			MMLU (↑)					MT-Bench (↑)
		Bio	*Cyber*	*Chem*	*College Bio*	*Virology*	*College CS*	*Cybersec*	*All*	
ZEPHYR-7B	Base	65.5	42.9	44.2	65.3	52.4	50.0	63.0	58.5	7.33
	LLMU	59.5	38.2	39.3	54.2	40.3	45.0	50.0	45.2	1.00
	SCRUB	43.4	37.3	39.6	53.5	41.0	49.0	62.0	51.9	7.09
	SSD	55.2	34.0	**35.9**	48.6	40.4	38.0	55.0	41.5	5.48
	CUT (ours)	**29.3**	**24.9**	40.5	64.6	22.9	47.0	50.0	**57.0**	**7.20**
YI-34B	Base	76.3	45.8	54.4	88.2	56.6	67.0	84.0	72.9	7.65
	CUT (ours)	30.9	29.2	50.2	81.9	27.7	52.0	46.0	69.0	7.11
GPT-4	Base	81.6	51.8	59.7	93.9	58.2	69.0	84.5	83.4	9.13

Table 4.1: Comparing base models and unlearning methods on question-answer evaluation (WMDP, MMLU) and fluency (MT-Bench). All WMDP and MMLU scores are percentage points. All unlearning methods were applied on removing WMDP-Bio and WMDP-Cyber.

4.3.2.2 Probing Results

In Figure 4.2, we evaluate the capability of probes to retrieve knowledge that has been retained by the model following the application of CUT. Our findings indicate that the accuracy achieved through linear probing marginally surpasses that of random chance. This inability of linear probes to discern the information to be removed implies that CUT does more than superficially conceal or obscure the data. Instead, it significantly modifies the model in a manner that obstructs the re-access of information that has been unlearned.

4.4 Related Work

Unlearning (Cao and Yang, 2015) originally gained traction as a response to privacy concerns in light of regulation (Council of European Union, 2018; California State Leglisature, 2018), and most methods focused on erasing specific samples or facts (Golatkar et al., 2020b; Ma et al., 2023; Meng et al., 2022; Jang et al., 2023; Pawelczyk et al., 2023) rather than entire domains. Goel et al. (2024) show existing unlearning methods struggle to remove knowledge without access to all relevant training data, a challenge CUT overcomes.

More recent methods erase broader concepts such as gender (Belrose et al., 2023), harmful behaviors (Yao et al., 2023b; Liu et al., 2024), or fictional universes (Eldan and Russinovich, 2023), but have not been proven to eliminate scientific knowledge which enables malicious use. Furthermore, most benchmarks for unlearning involve removing specific data samples (Google, 2023) or artificially chosen deletion sets (Choi and Na, 2023; Goel et al., 2023; Maini et al., 2024; Goel et al., 2024). In contrast, we study the removal of real-world information that enables malicious use.

There are other complementary strategies for improving safety against malicious use. These include input filtering (Inan et al., 2023) and learning from human preference data (Ziegler et al., 2020; Rafailov

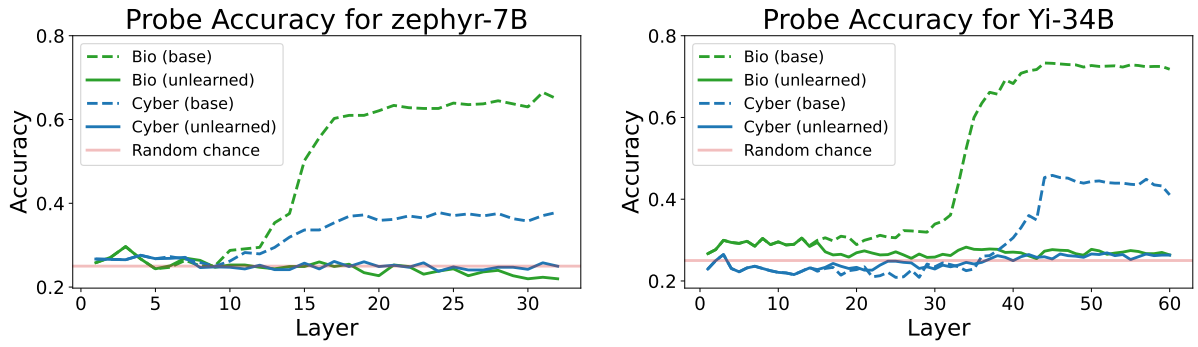

Figure 4.2: Linear probes cannot recover hazardous knowledge erased using CUT as the probe accuracy on unlearned models is random-chance. This indicates CUT also scrubs knowledge in model internals, not just outputs.

et al., 2023). However, these methods can be vulnerable to jailbreaks (Wei et al., 2023; Chao et al., 2023; Yao et al., 2023a; Yuan et al., 2023) and adversarial attacks (Wallace et al., 2019; Guo et al., 2021; Jones et al., 2023; Zou et al., 2023b). Another complementary approach is to remove hazardous data prior to pretraining (Ngo et al., 2021). However, knowledge and capabilities can get through this process, or even be introduced via later finetuning. We propose the use of unlearning as a post-hoc intervention to remove dangerous knowledge.

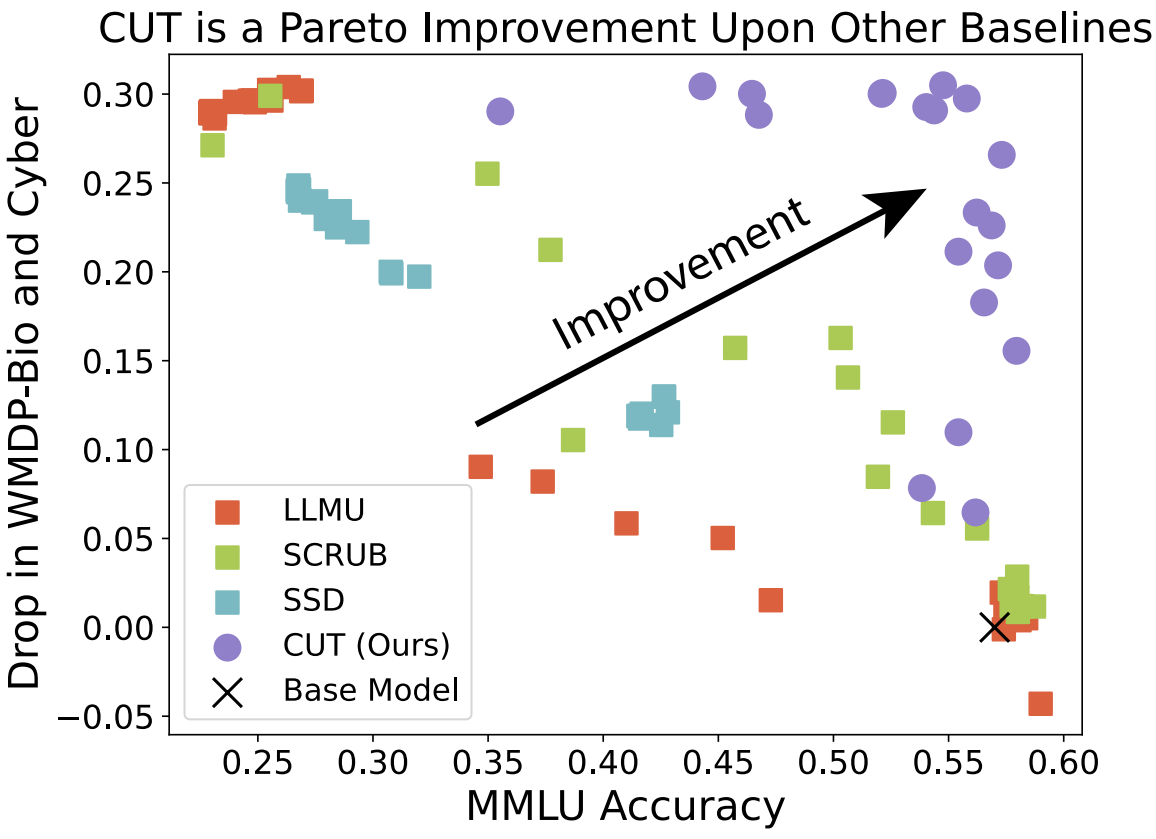

Figure 4.3: Results across a hyperparameter search. Compared to the other baselines, CUT is most capable of reducing WMDP performance while maintaining accuracy on MMLU.

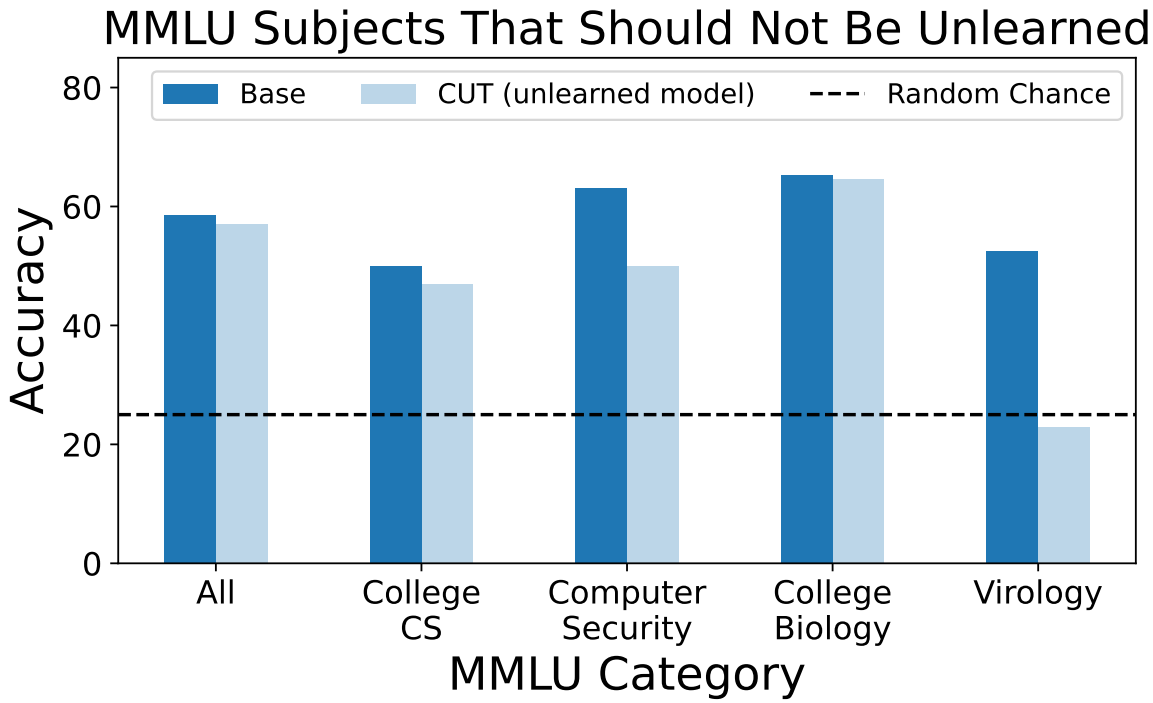

Figure 4.4: MMLU accuracy of ZEPHYR-7B with CUT. CUT preserves general biology and computer science knowledge. However, it unlearns too much: it removes introductory virology and computer security knowledge, indicating the scope for developing more surgical unlearning methods.

Chapter *5*

Discussion

5.1 Conclusion

While prior unlearning methods claim to handle arbitrary deletion sets, we prove that passing prior evaluations based on weight and output similarity fail to guarantee unlearning of non-IID deletion sets. This motivates the need for adversarial evaluations like our proposed Interclass Confusion test. In contrast to prior evaluations, the IC test is necessary to pass to achieve model indistinguishability and is not sensitive to different training procedures. Even with a small fraction of data being manipulated, the IC test can reliably capture how well unlearning procedures remove memorization of deletion set samples and properties generalized from them – both are important for different applications. We propose EU-k and CF-k as strong unlearning baselines that scale to large deletion sets, enable analysis of how early in the network information is retained and allow trading forgetting for efficiency at constant accuracy. We use our evaluation and methods to glean a variety of insights. (i) Unlearning methods that only modify the final layer in a deep network are not sufficient. (ii) We explore the interplay between learning and unlearning – theoretically, we conjecture that an unlearning procedure aiming to handle arbitrary deletions requires the ability to learn. Empirically, we show that better regularized models are more amenable to unlearning.

We hope that our analysis and proposed IC test along with EU-k and CF-k baselines will enable building stronger adversarial tests and better unlearning procedures. There is a need to bridge the current limitations of our work: We do not expect EU-k and CF-k to be gold-standard unlearning procedures, they are meant only as simple analytical tools that assist future research. As for the IC test, defining a passing score for real-world datasets that is necessary and sufficient is an open problem. While any procedure claiming to handle arbitrary deletions must pass the IC test, it alone cannot guarantee perfect unlearning. Finding a test that if passed is sufficient to prove unlearning of arbitrary deletions is an interesting direction.

We then treat the Corrective Machine Unlearning setting as a distinct requirement within Inexact Unlearning. Corrective Unlearning is designed to mitigate the negative effects of manipulated data

discovered post-training, such as diminished accuracy across specific domain areas, from an already trained model. This concept is grounded in an adversarial threat model, acknowledging that all the manipulated data samples may not be known. Instead, developers are often able to pinpoint only a representative subset of the manipulated samples.

Corrective Unlearning diverges significantly from the traditional privacy-oriented unlearning. Our findings indicate that latest unlearning methods, even the gold standard of retraining-from-scratch, fail to enhance accuracy on the manipulated domain unless nearly all of the manipulated data is identified. A notable exception is SSD (Foster et al., 2023), which successfully mitigates the effects of the BadNet (Gu et al., 2019) poison, thus illustrating the feasibility of removing the influence of manipulated data with only a small representative subset identified. However, this method does not work for the Interclass Confusion manipulation, which demonstrates the need for designing unlearning procedures that can ideally remove the influence of arbitrary manipulations. We hope our work spurs the development of stronger corrective unlearning methods and evaluations to assist practitioners in dealing with data quality issues arising from web-scale training.

We then demonstrate the use of unlearning to remove dual-use knowledge from Large Language Models, towards mitigating AI misuse by malicious actors. Specifically, we discuss CUT, a Representation Engineering based approach that steers models towards novice knowledge on specific domains. In the setting of knowledge removal from LLMs using a different representative corpus of samples compared to the training data, CUT outperforms adaptations of state of the art unlearning methods. We also propose the use of probing to test whether knowledge has really been unlearned from the internals of a model, in contrast to just producing random outputs on domain inputs.

Overall, as Machine Learning training runs become more expensive, it is less feasible to re-train new models every time an issue is detected. In this thesis, we have shown how Machine Unlearning can be used to remove the influence of undesirable training data or knowledge post-hoc. We hope unlearning can act as an effective complementary tool to data-filtering methods before training, as it acts to correct any oversights in the filtering process. An important challenge in unlearning is evaluating whether a model has really unlearnt, and we hope the contributions of this thesis, such as the Interclass Confusion Test, the Corrective Unlearning Paradigm, and Probing Evaluations, help ensure the robustness of future unlearning methods.

5.2 Limitations and Future Work

1) Adaptive Testing: The ideal corrective unlearning approaches should exhibit robustness against a broad spectrum of manipulation types. Specifically, these methods should withstand adaptive attacks, where the manipulations targeted for unlearning are crafted with knowledge of the unlearning procedures themselves (Tramer et al., 2020), not just the two evaluations we study. Similar to other related fields like adversarial robustness and privacy, it is important to design new Corrective Unlearning algorithms that work against powerful adaptive attacks.

2) Diverse Corrective Unlearning Evaluations: In addition, there is scope to design stronger evaluation frameworks for corrective unlearning. Apart from manipulating features and labels, adversaries could generate entirely synthetic samples (Zhang et al., 2019; Huang et al., 2020a). Although our focus is on supervised image classification, the concept of manipulation and its correction is also relevant in self-supervised learning contexts, such as language modeling (Wallace et al., 2020). Finally, an additional complexity could be the presence of false positives, where a clean sample getting identified as manipulated.

3) Theoretical Characterization of Corrective Unlearning: Current unlearning procedures aim to achieve a model distribution that is indistinguishably close to one obtained by retraining without certain samples, measured in terms like (ϵ, δ)-certified unlearning (Sekhari et al., 2021). However, we anticipate that the corrective unlearning problem will pave the way for innovative theoretical research. A critical area of interest is determining what conditions make a small 'representative set' of manipulated samples sufficient for effective corrective unlearning. Additionally, for a given manipulation class and a small set of such samples, it would be interesting to develop algorithms that prioritize improving accuracy on the manipulated domain over strict distributional indistinguishability. Another future challenge is to identify additional manipulated samples based on a small initial representative set.

4) Better evaluations and methods for LLM Unlearning: CUT shows how not having access to all the samples to be deleted can still be sufficient for unlearning, atleast in Large Language Models. However, CUT while good at reducing accuracy on the target domain to random chance, also has ripple effects on related domains of knowledge which may be desirable and not harmful. Future work can grapple with where to draw the boundary of what knowledge is potentially harmful, and how to ensure unlearning methods better adhere to these boundaries. Finally, more work is needed to ensure the unlearnt knowledge cannot be recovered easily, essentially measuring the robustness of unlearning (Lynch et al., 2024).

Related Publications

1. **Corrective Machine Unlearning**, *Shashwat Goel*, Ameya Prabhu, Philip Torr, Ponnurangam Kumaraguru, Amartya Sanyal. *Accepted at the Data-Centric Machine Learning (DMLR) Workshop at the 12th International Conference of Learning Representations (ICLR) 2024.*

2. **Corrective Machine Unlearning**, *Shashwat Goel*, Ameya Prabhu, Philip Torr, Ponnurangam Kumaraguru, Amartya Sanyal. *Under Review at the 4th Conference on Lifelong Learning Agents (CoLLAs) 2024*.

3. **The WMDP Benchmark: Measuring and Reducing Malicious Use With Unlearning**, Nathaniel Li, Alexander Pan, Anjali Gopal, Summer Yue, Daniel Berrios, Alice Gatti, Justin D. Li, Ann-Kathrin Dombrowski, *Shashwat Goel*, Long Phan, Gabriel Mukobi, Nathan Helm-Burger, Rassin Lababidi, Lennart Justen, Andrew B. Liu, Michael Chen, Isabelle Barrass, Oliver Zhang, Xiaoyuan Zhu, Rishub Tamirisa, Bhrugu Bharathi, Adam Khoja, Ariel Herbert-Voss, Cort B. Breuer, Andy Zou, Mantas Mazeika, Zifan Wang, Palash Oswal, Weiran Liu, Adam A. Hunt, Justin Tienken-Harder, Kevin Y. Shih, Kemper Talley, John Guan, Russell Kaplan, Ian Steneker, David Campbell, Brad Jokubaitis, Alex Levinson, Jean Wang, William Qian, Kallol Krishna Karmakar, Steven Basart, Stephen Fitz, Mindy Levine, Ponnurangam Kumaraguru, Uday Tupakula, Vijay Varadharajan, Yan Shoshitaishvili, Jimmy Ba, Kevin M. Esvelt, Alexandr Wang, Dan Hendrycks *Under Review at the 41st International Conference on Machine Learning (ICML) 2024*.

4. **Towards Adversarial Evaluations for Inexact Machine Unlearning.**, *Shashwat Goel*, Ameya Prabhu, Amartya Sanyal, Ser-Nam Lim, Philip Torr, Ponnurangam Kumaraguru. *arXiv preprint arXiv:2201.06640 (2023)*.

Other Publications

1. **Probing Negation in Language Models.**, Shashwat Singh, *Shashwat Goel*, Saujas Vaduguru, Ponnurangam Kumaraguru *8th Workshop on Representation Learning for NLP (RepL4NLP) at the 61st Annual Meeting of the Association for Computational Linguistics (ACL) 2023*.

2. **Proportional Aggregation of Preferences for Sequential Decision Making.**, Nikhil Chandak, *Shashwat Goel*, Dominik Peters. *Outstanding Paper Award (top 3 out of 12,000+ submissions)*

at the 38th Annual Conference of the Association for the Advancement of Artificial Intelligence (AAAI) 2024.

3. **Representation Engineering: A Top-Down Approach to AI Transparency.**, Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, *Shashwat Goel*, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, Dan Hendrycks. *arXiv preprint arXiv:2310.01405 (2023)*.

Bibliography

(1982). Alternative k-nearest neighbour rules in supervised pattern recognition: Part 1. k-nearest neighbour classification by using alternative voting rules. *Analytica Chimica Acta*.

(2018). Humans forget, machines remember: Artificial intelligence and the right to be forgotten. *Computer Law & Security Review*.

01-ai (2023). GitHub - 01-ai/Yi: A series of large language models trained from scratch by developers @01-ai — github.com. https://github.com/01-ai/Yi.

Acquisti, A., Friedman, A., and Telang, R. (2006). Is there a cost to privacy breaches? an event study.

Aldaghri, N., Mahdavifar, H., and Beirami, A. (2021). Coded machine unlearning. In *IEEE Access*.

Anthropic (2023). Anthropic's Responsible Scaling Policy — anthropic.com. https://www.anthropic.com/index/anthropics-responsible-scaling-policy.

Banko, M. and Brill, E. (2001). Scaling to very very large corpora for natural language disambiguation. In *Annual Meeting of the Association for Computational Linguistics*.

Barreno, M., Nelson, B., Sears, R., Joseph, A. D., and Tygar, J. D. (2006). Can machine learning be secure? In *ASIA Conference on Computer and Communications Security (ACM ASIACCS)*.

Bau, D., Liu, S., Wang, T., Zhu, J.-Y., and Torralba, A. (2020). Rewriting a deep generative model. In *European Conference on Computer Vision*.

Baumhauer, T., Schöttle, P., and Zeppelzauer, M. (2022). Machine unlearning: linear filtration for logit-based classifiers. *Machine Learning*.

Belrose, N., Schneider-Joseph, D., Ravfogel, S., Cotterell, R., Raff, E., and Biderman, S. (2023). Leace: Perfect linear concept erasure in closed form. *NeurIPS*.

Bhatt, M., Chennabasappa, S., Nikolaidis, C., Wan, S., Evtimov, I., Gabi, D., Song, D., Ahmad, F., Aschermann, C., Fontana, L., Frolov, S., Giri, R. P., Kapil, D., Kozyrakis, Y., LeBlanc, D., Milazzo, J., Straumann, A., Synnaeve, G., Vontimitta, V., Whitman, S., and Saxe, J. (2023). Purple llama cyberseceval: A secure coding benchmark for language models.

Boiko, D. A., MacKnight, R., Kline, B., and Gomes, G. (2023). Autonomous chemical research with large language models. *Nature*, 624(7992):570–578.

Bourtoule, L., Chandrasekaran, V., Choquette-Choo, C. A., Jia, H., Travers, A., Zhang, B., Lie, D., and Papernot, N. (2021). Machine unlearning. In *IEEE Symposium on Security and Privacy (SP)*.

Breck, E., Zinkevich, M., Polyzotis, N., Whang, S., and Roy, S. (2019). Data validation for machine learning. In *Proceedings of SysML*.

Brock, A., Lim, T., Ritchie, J. M., and Weston, N. (2017). Freezeout: Accelerate training by progressively freezing layers. *arxiv:1706.04983*.

Brodley, C. E. and Friedl, M. A. (1999). Identifying mislabeled training data. *Journal of Artifical Intelligence Research (JAIR)*.

Brophy, J. and Lowd, D. (2021). Machine unlearning for random forests. In *International Conference on Machine Learning (ICML)*.

Buolamwini, J. and Gebru, T. (2018). Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*.

California State Leglisature (2018). California consumer privacy act.

Cao, Y. and Yang, J. (2015). Towards making systems forget with machine unlearning. In *IEEE Symposium on Security and Privacy (SP)*.

Carlini, N., Jagielski, M., Choquette-Choo, C. A., Paleka, D., Pearce, W., Anderson, H., Terzis, A., Thomas, K., and Tramèr, F. (2023). Poisoning web-scale training datasets is practical. In *IEEE Symposium on Security and Privacy (SP)*.

Carlini, N., Liu, C., Erlingsson, Ú., Kos, J., and Song, D. (2019). The secret sharer: Evaluating and testing unintended memorization in neural networks. In *USENIX*.

Cauwenberghs, G. and Poggio, T. (2000). Incremental and decremental support vector machine learning. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Chao, P., Robey, A., Dobriban, E., Hassani, H., Pappas, G. J., and Wong, E. (2023). Jailbreaking black box large language models in twenty queries.

Chen, H., Fu, C., Zhao, J., and Koushanfar, F. (2019). Deepinspect: A black-box trojan detection and mitigation framework for deep neural networks. In *International Joint Conference on Artificial Intelligence*.

Chen, J. and Yang, D. (2023). Unlearn what you want to forget: Efficient unlearning for LLMs. In *Conference on Empirical Methods in Natural Language Processing*.

Chen, M., Zhang, Z., Wang, T., Backes, M., Humbert, M., and Zhang, Y. (2021). When machine unlearning jeopardizes privacy.

Chen, X., Liu, C., Li, B., Lu, K., and Song, D. (2017). Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv:1712.05526*.

Choi, D. and Na, D. (2023). Towards machine unlearning benchmarks: Forgetting the personal identities in facial recognition systems.

Chourasia, R., Shah, N., and Shokri, R. (2023). Forget unlearning: Towards true data-deletion in machine learning. *International Conference on Learning Representations (ICLR)*.

Chundawat, V. S., Tarun, A. K., Mandal, M., and Kankanhalli, M. (2023a). Can bad teaching induce forgetting? unlearning in deep networks using an incompetent teacher. In *Annual AAAI Conference on Artificial Intelligence*.

Chundawat, V. S., Tarun, A. K., Mandal, M., and Kankanhalli, M. (2023b). Zero-shot machine unlearning. *IEEE Transactions on Information Forensics and Security*.

Council of European Union (2018). Eu general data protection regulation.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. (2006). Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography (TOC*.

Eldan, R. and Russinovich, M. (2023). Who's harry potter? approximate unlearning in llms. *arXiv:2310.02238*.

Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., DasSarma, N., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. (2021). A mathematical framework for transformer circuits. *Transformer Circuits Thread*.

Fabbrizzi, S., Papadopoulos, S., Ntoutsi, E., and Kompatsiaris, I. (2021). A survey on bias in visual datasets. *arXiv:2107.07919*.

Fang, R., Bindu, R., Gupta, A., Zhan, Q., and Kang, D. (2024). Llm agents can autonomously hack websites.

Federal Trade Commission (2021). California company settles ftc allegations it deceived consumers about use of facial recognition in photo storage app.

Feldman, V. and Zhang, C. (2020). What neural networks memorize and why: Discovering the long tail via influence estimation. *Conference on Neural Information Processing Systems (NeurIPS)*.

Foster, J., Schoepf, S., and Brintrup, A. (2023). Fast machine unlearning without retraining through selective synaptic dampening. *arXiv:2308.07707*.

Fowl, L., Goldblum, M., Chiang, P.-y., Geiping, J., Czaja, W., and Goldstein, T. (2021). Adversarial examples make strong poisons. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Fredrikson, M., Jha, S., and Ristenpart, T. (2015). Model inversion attacks that exploit confidence information and basic countermeasures.

Frenay, B. and Verleysen, M. (2014). Classification in the presence of label noise: A survey. In *IEEE Transactions on Neural Networks and Learning Systems*.

French, R. M. (1999). Catastrophic forgetting in connectionist networks. In *Trends in Cognitive Sciences*.

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., et al. (2020). The pile: An 800gb dataset of diverse text for language modeling. *arXiv:2101.00027*.

Gao, L., Tow, J., Biderman, S., Black, S., DiPofi, A., Foster, C., Golding, L., Hsu, J., McDonell, K., Muennighoff, N., Phang, J., Reynolds, L., Tang, E., Thite, A., Wang, B., Wang, K., and Zou, A. (2021). A framework for few-shot language model evaluation.

Ginart, A., Guan, M. Y., Valiant, G., and Zou, J. (2019). Making AI forget you: Data deletion in machine learning. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Goel, S., Prabhu, A., Sanyal, A., Lim, S.-N., Torr, P., and Kumaraguru, P. (2023). Towards adversarial evaluations for inexact machine unlearning. *arXiv:2201.06640*.

Goel, S., Prabhu, A., Torr, P., Kumaraguru, P., and Sanyal, A. (2024). Corrective machine unlearning.

Golatkar, A., Achille, A., Ravichandran, A., Polito, M., and Soatto, S. (2021). Mixed-privacy forgetting in deep networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

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

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

Goldblum, M., Tsipras, D., Xie, C., Chen, X., Schwarzschild, A., Song, D., Madry, A., Li, B., and Goldstein, T. (2022). Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Google (2023). Neurips 2023 machine unlearning challenge.

Gopal, A., Helm-Burger, N., Justen, L., Soice, E. H., Tzeng, T., Jeyapragasan, G., Grimm, S., Mueller, B., and Esvelt, K. M. (2023). Will releasing the weights of future large language models grant widespread access to pandemic agents?

Graves, L., Nagisetty, V., and Ganesh, V. (2021). Amnesiac machine learning. In *Annual AAAI Conference on Artificial Intelligence*.

Grosse, R., Bae, J., Anil, C., Elhage, N., Tamkin, A., Tajdini, A., Steiner, B., Li, D., Durmus, E., Perez, E., et al. (2023). Studying large language model generalization with influence functions. *arXiv:2308.03296*.

Gu, T., Liu, K., Dolan-Gavitt, B., and Garg, S. (2019). Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*.

Guo, C., Goldstein, T., Hannun, A., and van der Maaten, L. (2020). Certified data removal from machine learning models. In *International Conference on Machine Learning (ICML)*.

Guo, C., Sablayrolles, A., Jégou, H., and Kiela, D. (2021). Gradient-based adversarial attacks against text transformers.

Gupta, V., Jung, C., Neel, S., Roth, A., Sharifi-Malvajerdi, S., and Waites, C. (2021). Adaptive machine unlearning. *Conference on Neural Information Processing Systems (NeurIPS)*.

Han, X., Xu, G., Zhou, Y., Yang, X., Li, J., and Zhang, T. (2022). Physical backdoor attacks to lane detection systems in autonomous driving. In *ACM International Conference on Multimedia*.

Hardt, M., Price, E., and Srebro, N. (2016). Equality of opportunity in supervised learning. *Conference on Neural Information Processing Systems (NeurIPS)*.

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

He, Y., Meng, G., Chen, K., He, J., and Hu, X. (2021). Deepobliviate: A powerful charm for erasing data residual memory in deep neural networks. *arXiv:2105.06209*.

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. (2020). Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Hooker, S. (2021). Moving beyond "algorithmic bias is a data problem". *Patterns*.

Hu, H., Salcic, Z., Dobbie, G., and Zhang, X. (2021). Membership inference attacks on machine learning: A survey. *arXiv:2103.07853*.

Huang, W. R., Geiping, J., Fowl, L., Taylor, G., and Goldstein, T. (2020a). Metapoison: Practical general-purpose clean-label data poisoning. *Conference on Neural Information Processing Systems (NeurIPS)*.

Huang, Y., Song, Z., Li, K., and Arora, S. (2020b). Instahide: Instance-hiding schemes for private distributed learning. In *International Conference on Machine Learning (ICML)*.

Idelbayev, Y. (2018). Proper ResNet implementation for CIFAR10/CIFAR100 in PyTorch.

Ilharco, G., Ribeiro, M. T., Wortsman, M., Schmidt, L., Hajishirzi, H., and Farhadi, A. (2023). Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*.

Inan, H., Upasani, K., Chi, J., Rungta, R., Iyer, K., Mao, Y., Tontchev, M., Hu, Q., Fuller, B., Testuggine, D., and Khabsa, M. (2023). Llama guard: Llm-based input-output safeguard for human-ai conversations.

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

Jagielski, M., Oprea, A., Biggio, B., Liu, C., Nita-Rotaru, C., and Li, B. (2018). Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *IEEE Symposium on Security and Privacy (SP)*.

Jang, J., Yoon, D., Yang, S., Cha, S., Lee, M., Logeswaran, L., and Seo, M. (2023). Knowledge unlearning for mitigating privacy risks in language models. In *Annual Meeting of the Association for Computational Linguistics*.

Jiang, H. and Nachum, O. (2020). Identifying and correcting label bias in machine learning. In *International Conference on Artificial Intelligence and Statistics*.

Jones, E., Dragan, A., Raghunathan, A., and Steinhardt, J. (2023). Automatically auditing large language models via discrete optimization.

Kataoka, H., Okayasu, K., Matsumoto, A., Yamagata, E., Yamada, R., Inoue, N., Nakamura, A., and Satoh, Y. (2020). Pre-training without natural images.

Konstantinov, N. H. and Lampert, C. (2022). Fairness-aware pac learning from corrupted data. *Journal of Machine Learning Research (JMLR)*.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*.

Kurmanji, M., Triantafillou, P., and Triantafillou, E. (2023). Towards unbounded machine unlearning. *Conference on Neural Information Processing Systems (NeurIPS)*.

Li, J. and Ghosh, S. (2023). Random relabeling for efficient machine unlearning. *arXiv:2305.12320*.

Li, N., Pan, A., Gopal, A., Yue, S., Berrios, D., Gatti, A., Li, J. D., Dombrowski, A.-K., Goel, S., Phan, L., Mukobi, G., Helm-Burger, N., Lababidi, R., Justen, L., Liu, A. B., Chen, M., Barrass, I., Zhang, O., Zhu, X., Tamirisa, R., Bharathi, B., Khoja, A., Herbert-Voss, A., Breuer, C. B., Zou, A., Mazeika, M., Wang, Z., Oswal, P., Liu, W., Hunt, A. A., Tienken-Harder, J., Shih, K. Y., Talley, K., Guan, J., Kaplan, R., Steneker, I., Campbell, D., Jokubaitis, B., Levinson, A., Wang, J., Qian, W., Karmakar, K. K., Basart, S., Fitz, S., Levine, M., Kumaraguru, P., Tupakula, U., Varadharajan, V., Shoshitaishvili,

Y., Ba, J., Esvelt, K. M., Wang, A., and Hendrycks, D. (2024). The wmdp benchmark: Measuring and reducing malicious use with unlearning.

Li, Y., Wu, B., Jiang, Y., Li, Z., and Xia, S.-T. (2020). Backdoor learning: A survey. *arXiv:2007.08745*.

Liu, Z., Dou, G., Tan, Z., Tian, Y., and Jiang, M. (2024). Towards Safer Large Language Models through Machine Unlearning. *arXiv e-prints*, page arXiv:2402.10058.

Lynch, A., Guo, P., Ewart, A., Casper, S., and Hadfield-Menell, D. (2024). Eight methods to evaluate robust unlearning in llms.

Ma, Z., Liu, Y., Liu, X., Liu, J., Ma, J., and Ren, K. (2023). Learn to forget: Machine unlearning via neuron masking. *IEEE Transactions on Dependable and Secure Computing*.

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

Maini, P., Feng, Z., Schwarzschild, A., Lipton, Z. C., and Kolter, J. Z. (2024). Tofu: A task of fictitious unlearning for llms.

Mehta, R., Pal, S., Singh, V., and Ravi, S. N. (2022). Deep unlearning via randomized conditionally independent hessians. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Meng, K., Bau, D., Andonian, A., and Belinkov, Y. (2022). Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, 35.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. (2016). Pointer sentinel mixture models.

Mitchell, E., Lin, C., Bosselut, A., Manning, C. D., and Finn, C. (2022). Memory-based model editing at scale. In *International Conference on Machine Learning (ICML)*.

Nakkiran, P. (2019). A discussion of 'adversarial examples are not bugs, they are features': Adversarial examples are just bugs, too. *Distill*.

Nakkiran, P. and Bansal, Y. (2020). Distributional generalization: A new kind of generalization. *arxiv:2009.08092*.

Neel, S., Roth, A., and Sharifi-Malvajerdi, S. (2021). Descent-to-delete: Gradient-based methods for machine unlearning. In *Conference on Learning Theory (COLT)*.

Ngo, H., Raterink, C. D., de Ara'ujo, J. M., Zhang, I., Chen, C., Morisot, A., and Frosst, N. (2021). Mitigating harm in language models with conditional-likelihood filtration. *ArXiv*, abs/2108.07790.

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

Northcutt, C. G., Athalye, A., and Mueller, J. (2021a). Pervasive label errors in test sets destabilize machine learning benchmarks. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Northcutt, C. G., Jiang, L., and Chuang, I. L. (2021b). Confident learning: Estimating uncertainty in dataset labels. In *Journal of Artifical Intelligence Research (JAIR)*.

OpenAI (2023a). Gpt-4 technical report.

OpenAI (2023b). Preparedness — openai.com. https://openai.com/safety/preparedness.

OpenAI (2024). Building an early warning system for LLM-aided biological threat creation — openai.com. https://openai.com/research/building-an-early-warning-system-for-llm-aided-biological-threat-creation.

Paleka, D. and Sanyal, A. (2023). A law of adversarial risk, interpolation, and label noise. In *International Conference on Learning Representations (ICLR)*.

Parliament of Canada (2018). Personal information protection and electronic documents act.

Pawelczyk, M., Neel, S., and Lakkaraju, H. (2023). In-context unlearning: Language models as few shot unlearners. *arXiv preprint arXiv:2310.07579*.

Peste, A., Alistarh, D., and Lampert, C. H. (2021). Ssse: Efficiently erasing samples from trained machine learning models. In *NeurIPS Workshop Privacy in Machine Learning*.

Pleiss, G., Zhang, T., Elenberg, E., and Weinberger, K. Q. (2020). Identifying mislabeled data using the area under the margin ranking. *Conference on Neural Information Processing Systems (NeurIPS)*.

Prabhu, V. U. and Birhane, A. (2021). Large image datasets: A pyrrhic win for computer vision? In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)* .

Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C. D., and Finn, C. (2023). Direct preference optimization: Your language model is secretly a reward model.

Sandbrink, J. B. (2023). Artificial intelligence and biological misuse: Differentiating risks of language models and biological design tools.

Sanyal, A., Dokania, P. K., Kanade, V., and Torr, P. (2021). How benign is benign overfitting? In *International Conference on Learning Representations (ICLR)*.

Sanyal, A., Hu, Y., and Yang, F. (2022). How unfair is private learning? In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*.

Schelter, S. (2020). “amnesia” - machine learning models that can forget user data very fast. In *Conference on Innovative Data Systems Research (CIDR)*.

Schelter, S., Grafberger, S., and Dunning, T. (2021). Hedgecut: Maintaining randomised trees for low-latency machine unlearning. In *Symposium on Principles of Database Systems (SIGMOD/PODS)*.

Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., et al. (2022). Laion-5b: An open large-scale dataset for training next generation image-text models. *Conference on Neural Information Processing Systems (NeurIPS)*.

Sclar, M., Choi, Y., Tsvetkov, Y., and Suhr, A. (2023). Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting.

Sekhari, A., Acharya, J., Kamath, G., and Suresh, A. T. (2021). Remember what you want to forget: Algorithms for machine unlearning. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Shibata, T., Irie, G., Ikami, D., and Mitsuzumi, Y. (2021). Learning with selective forgetting. In *International Joint Conference on Artificial Intelligence*.

Shokri, R., Stronati, M., Song, C., and Shmatikov, V. (2017). Membership inference attacks against machine learning models. In *IEEE Symposium on Security and Privacy (SP)*.

Shorten, C. and Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*.

Sommer, D. M., SOng, L., Wagh, S., and Mittal, P. (2022). Athena: Probabilistic Verification of Machine Unlearning. *Proceedings on Privacy Enhancing Technologies (PoPETS)*.

Song, H., Kim, M., Park, D., Shin, Y., and Lee, J.-G. (2022). Learning from noisy labels with deep neural networks: A survey. *IEEE Transactions on Neural Networks and Learning Systems*.

Song, L. and Mittal, P. (2021). Systematic evaluation of privacy risks of machine learning models. In *USENIX*.

Tarun, A. K., Chundawat, V. S., Mandal, M., and Kankanhalli, M. (2023). Fast yet effective machine unlearning. *IEEE Transactions on Neural Networks and Learning Systems*.

Thudi, A., Deza, G., Chandrasekaran, V., and Papernot, N. (2021). Unrolling sgd: Understanding factors influencing machine unlearning. *arXiv preprint arXiv:2109.13398*.

Thudi, A., Jia, H., Shumailov, I., and Papernot, N. (2022). On the necessity of auditable algorithmic definitions for machine unlearning. In *USENIX*.

Tian, Z., Cui, L., Liang, J., and Yu, S. (2022). A comprehensive survey on poisoning attacks and countermeasures in machine learning. *ACM Computing Surveys*.

Tramer, F., Carlini, N., Brendel, W., and Madry, A. (2020). On adaptive attacks to adversarial example defenses. *Conference on Neural Information Processing Systems (NeurIPS)*.

Tsai, C.-H., Lin, C.-Y., and Lin, C.-J. (2014). Incremental and decremental training for linear classification.

Tunstall, L., Beeching, E., Lambert, N., Rajani, N., Rasul, K., Belkada, Y., Huang, S., von Werra, L., Fourrier, C., Habib, N., Sarrazin, N., Sanseviero, O., Rush, A. M., and Wolf, T. (2023). Zephyr: Direct distillation of lm alignment.

Turk, M. and Pentland, A. (1991). Eigenfaces for recognition. *Journal of cognitive neuroscience*.

Turner, A. M., Thiergart, L., Udell, D., Leech, G., Mini, U., and MacDiarmid, M. (2023). Activation addition: Steering language models without optimization.

UK Cabinet Office (2023). National risk register. Technical report, UK Cabinet Office.

Wallace, E., Feng, S., Kandpal, N., Gardner, M., and Singh, S. (2019). Universal adversarial triggers for attacking and analyzing nlp. *arXiv preprint arXiv:1908.07125*.

Wallace, E., Zhao, T. Z., Feng, S., and Singh, S. (2020). Concealed data poisoning attacks on nlp models. *arXiv:2010.12563*.

Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H., and Zhao, B. Y. (2019). Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *IEEE Symposium on Security and Privacy (SP)*.

Warnecke, A., Pirch, L., Wressnegger, C., and Rieck, K. (2021). Machine unlearning of features and labels. *Network and Distributed System Security Symposium*.

Wei, A., Haghtalab, N., and Steinhardt, J. (2023). Jailbroken: How does llm safety training fail? *arXiv preprint arXiv:2307.02483*.

White House, T. (2023). Executive Order on the Safe, Secure, and Trustworthy Development and Use of Artificial Intelligence. https://www.whitehouse.gov/briefing-room/presidential-actions/2023/10/30/executive-order-on-the-safe-secure-and-trustworthy-development-and-use-of-artif

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

Yan, H., Li, X., Guo, Z., Li, H., Li, F., and Lin, X. (2022). Arcane: An efficient architecture for exact machine unlearning. In *International Joint Conference on Artificial Intelligence*.

Yang, C., Wu, Q., Li, H., and Chen, Y. (2020). Generative poisoning attack method against neural networks. *arXiv:1703.01340*.

Yao, D., Zhang, J., Harris, I. G., and Carlsson, M. (2023a). Fuzzllm: A novel and universal fuzzing framework for proactively discovering jailbreak vulnerabilities in large language models.

Yao, Y., Xu, X., and Liu, Y. (2023b). Large language model unlearning. *arXiv:2310.10683*.

Yosinski, J., Clune, J., Fuchs, T., and Lipson, H. (2015). Understanding neural networks through deep visualization.

Yuan, Y., Jiao, W., Wang, W., tse Huang, J., He, P., Shi, S., and Tu, Z. (2023). Gpt-4 is too smart to be safe: Stealthy chat with llms via cipher.

Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., and Yoo, Y. (2019). Cutmix: Regularization strategy to train strong classifiers with localizable features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zagoruyko, S. and Komodakis, N. (2016). Wide residual networks. In *The British Machine Vision Conference*.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2017). Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations (ICLR)*.

Zhang, J., Chen, J., Wu, D., Chen, B., and Yu, S. (2019). Poisoning attack in federated learning using generative adversarial nets. In *IEEE international conference on big data science and engineering (TrustCom/BigDataSE)*.

Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E. P., Zhang, H., Gonzalez, J. E., and Stoica, I. (2023). Judging llm-as-a-judge with mt-bench and chatbot arena.

Ziegler, D. M., Stiennon, N., Wu, J., Brown, T. B., Radford, A., Amodei, D., Christiano, P., and Irving, G. (2020). Fine-tuning language models from human preferences.

Zou, A., Phan, L., Chen, S., Campbell, J., Guo, P., Ren, R., Pan, A., Yin, X., Mazeika, M., Dombrowski, A.-K., et al. (2023a). Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*.

Zou, A., Wang, Z., Kolter, J. Z., and Fredrikson, M. (2023b). Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*.