

A Additional Related Work

Machine Unlearning. Techniques like data sharding [Bourtoule et al., 2021, Chen et al., 2022] partition the training process in such a way that only a portion of the model needs to be retrained when removing a subset of the dataset, reducing the computational burden compared to retraining the entire model. For example, [Guo et al., 2020], [Neel et al., 2021], [Chien et al., 2023] analyzed the influence of removed data on linear or convex models and proposed gradient-based updates on model parameters to remove this influence. [Chourasia and Shah, 2023] proposed an unlearning method that appears similar to ours but differs in key aspects, particularly in the definition of advantage, a term whose meaning varies by threat model. Their threat model leads to a criterion for unlearning effectiveness, but like theory-based approaches such as certified removal, it is hard to empirically evaluate for most approximate methods lacking guarantees. In contrast, our threat model is tailored to enable a practical and novel evaluation metric, addressing a key gap in unlearning research.

Retraining-based Evaluation. Generally, retraining-based evaluation seeks to compare unlearned models to retrained models. As introduced in the works by [Golatkar et al., 2021], [He et al., 2021], [Golatkar et al., 2020], model accuracy on the forget set should be similar to the accuracy on the test set as if the forget set never exists in the training set. [Peste et al., 2021] proposed an evaluation metric based on normalized confusion matrix element-wise difference on selected data samples. [Golatkar et al., 2021] proposed using relearn time, which is the additional time to use for unlearned models to perform comparably to retrained models. The authors also proposed to measure the ℓ_1 distance between the final activations of the scrubbed weights and the retrained model. [Wu et al., 2020], [Izzo et al., 2021] turned to ℓ_2 distance of weight parameters between unlearned models and retrained models. In general, beyond the need for additional implementation and the lower computational efficiency inherent in retraining-based evaluations, a more critical issue is the influence of random factors. As discussed by [Cretu et al., 2023], such random factors, including the sequence of data batches and the initial configuration of models, can lead to the unaligned storage of information within models. This misalignment may foster implicit biases favoring certain retrained models.

Theory-based Evaluation. Some literature tries to characterize data removal efficacy by requiring a strict theoretical guarantee for the unlearned models. However, these methods have strong model assumptions, such as convexity or linearity, or require inefficient white-box access to target models, thus limiting their applicability in practice. For example, [Guo et al., 2020], [Neel et al., 2021], [Chien et al., 2023] focus on the notion of the certified removal (CR), which requires that the unlearned model cannot be statistically distinguished from the retrained model. By definition, CR is parametrized by privacy parameters called *privacy budgets*, which quantify the level of statistical indistinguishability. Hence, models with CR guarantees will intrinsically satisfy an “evaluation metric” induced by the definition of CR, acting as a form of “evaluation.” On the other hand, [Becker and Liebig, 2022] adopted an information-theoretical perspective and turned to epistemic uncertainty to evaluate the information remaining after unlearning.

Attack-based Evaluation. Since attacks are the most direct way to interpret privacy risks, attack-based evaluation is a common metric in unlearning literature. The classical approach is to directly calculate the MIA accuracy using various kinds of MIAs [Graves et al., 2020, Kurmanji et al., 2023]. One kind of MIA utilizes shadow models [Shokri et al., 2017], which are trained with the same model structure as the original models but on a shadow dataset sampled with the same data sampling distribution. Moreover, some MIAs calculate membership scores based on correctness and confidence [Song and Mittal, 2021]. Some evaluation metrics do move beyond the vanilla MIA accuracy. For example, [Triantafillou and Kairouz, 2023] leveraged hypothesis testing coupled with MIAs to compute an estimated privacy budget for each unlearning method, which gives a rather rigorous estimation of unlearning efficacy. [Hayes et al., 2024] proposed a novel MIA towards machine unlearning based on Likelihood Ratio Attack and evaluated machine unlearning through a combination of the predicted membership probability and the *balanced* MIA accuracy on test and forget sets. They designed a new MIA attack with a similar attack-defense game framework. There are other evaluation metrics also based on MIAs, but with different focuses. However, as they still use MIA accuracy as the evaluation metric, the game itself doesn’t bring much for their evaluation framework other than a clear experiment procedure. [Goel et al., 2023] proposed an Interclass Confusion (IC) test that manipulates the input dataset to evaluate both model indistinguishability and property generalization. However, their metric is less direct in terms of interpreting real-life privacy risks. Lastly, For example, [Chen et al., 2021] proposed a novel metric based on MIAs that know both learned and unlearned models with a focus on how much information is deleted rather than how much

information is left after the unlearning process. Sommer et al. [2020] provided a backdoor verification mechanism for Machine Learning as a Service (MLaaS), which benefits an individual user valuing his/her privacy to verify the efficacy of unlearning. They focus more on user-level verification rather than model-level evaluation.

B Omitted Details From Section 3

B.1 More details of Advantage

In cryptographic games, there are two interacting players, a benign player named *challenger* representing the cryptographic protocol under evaluation (corresponding to the unlearning algorithm in our context), and an *adversary* attempts to compromise the security properties of the challenger. The game proceeds in several phases, including an initialization phase where the game is initialized with specific configuration parameters, a challenger phase where the challenger performs the cryptographic protocol, and an adversary phase where the adversary queries allowed by the game’s rules and generates a guess of the secret protected by the challenger. Finally, the game concludes with a win or loss for the adversary, depending on their guess. In the context of machine unlearning, the goal of the adversary is to guess whether certain given data come from the set to be unlearned (the forget set) or the set never used in training (the test set), based on access to the unlearned model.

The notion of advantage quantifies, in probabilistic terms, how effectively an adversary can win the game when it is played repeatedly. It is often defined as the difference in the adversary’s accepting rate between two distinct scenarios (e.g., with or without access to information potentially leaked by the cryptographic protocol) [Katz and Lindell, 2007]. In the context of machine unlearning, the two scenarios can refer to the data given to the adversary coming from either the forget set or the test set, respectively. The game is constructed such that, if the cryptographic protocol is perfectly secure (i.e., the unlearned model has completely erased the information of the forget set), the adversary’s advantage is expected to be zero, making it a well-calibrated measure of the protocol’s security.

B.2 Design Choices

In this section, we justify some of our design choices when designing the unlearning sample inference game. Most of them are of practical consideration, while some are for the convenience of analysis.

Uniform Splitting. At the initialization phase, we choose the split uniformly rather than allowing sampling from an arbitrary distribution. The reason is two-fold: Firstly, since this sampling strategy corresponds to the so-called *i.i.d. unlearning* setup [Qu et al., 2024], i.e., the unlearned samples will be drawn from a distribution of \mathcal{D} in an i.i.d. fashion. In this regard, uniformly splitting the dataset corresponds to a uniform distribution of \mathcal{D} for the unlearned samples to be drawn from. This is the most commonly used sampling strategy when evaluating unlearning algorithms since it’s difficult to estimate the probability that data will be requested to be unlearned.

Secondly, [Qu et al., 2024] acknowledged the significantly greater difficulty of non-i.i.d. unlearning compared to i.i.d. unlearning empirically. A classic example of non-i.i.d. unlearning is the process of unlearning an entire class of data, where a subset of data shares the same label. Conversely, even non-uniform splitting complicates the analysis, leading to the breakdown of our theoretical results. Specifically, generalizing both Theorem 3.3 and Theorem 3.5 becomes non-trivial. Overall, non-uniform splitting presents obstacles both empirically and theoretically.

Sensitivity Distribution and Equal-Size Splits. The role of the sensitivity distribution $\mathbb{P}_{\mathcal{D}}$ is to capture biases stemming from various origins, such that more sensitive data will have greater sampling probability, hence greater privacy risks. For instance, if the forget set comprises data that users request to delete, with some being more sensitive than others, a corresponding bias should be incorporated into the game. In particular, we tailor our random oracle $\mathcal{O}_s(b)$ to sample data according to $\mathbb{P}_{\mathcal{D}}$, so when the adversary engages with the oracle, it gains increased exposure to more sensitive data, compelling the challenger to unlearn such data more effectively, thereby necessitating a heightened level of defense.

Accommodating $\mathbb{P}_{\mathcal{D}}$ also justifies restriction (b) when splitting the dataset \mathcal{D} . In particular, even if we enforce $|\mathcal{F}| = |\mathcal{T}|$, we still have the freedom of choosing the sensitivity distribution to have zero probability mass on some data points. In this regard, enforcing the restriction (b) is still general enough to capture unequal-size splits.

Intrinsic Learning Algorithm for Challenger. The challenger, which we denote as UL, *has a learning algorithm LR in mind* in our formulation. This is because the existing theory-based unlearning method, such as the certified removal algorithm [Guo et al., 2020] as defined in Definition B.3 is achieved by a combination of the learning algorithm and a corresponding unlearning method to support unlearning request with theoretical guarantees. In other words, given an arbitrary learning algorithm LR, it’s unlikely to design an efficient unlearning algorithm UL with strong theoretical guarantees, at least this is not captured by the notion of certified removal. Hence, allowing the challenger to choose its learning algorithm accommodates this situation.

Black-box Adversary v.s. White-box Adversary. By default, we assume that m is given to \mathcal{A} in a *black-box* fashion, i.e., \mathcal{A} only has oracle access to m . However, our framework can also adapt white-box adversaries which requires full model parameters of m . The only difference is that the efficiency definition changes accordingly, i.e., polynomial time in the size of $|\mathcal{D}|$ for a black-box adversary or polynomial time in the number of parameters of m for a white-box adversary.

Strong Adversary in Practice. As discussed in Section 3.5, the current state-of-the-art MIA adversaries are all *weak*. While it is possible to formulate the unlearning sample inference game entirely with the weak adversary, we discuss the rationale behind considering the strong adversary. One of the apparent reasons is simply because the strong adversary encompasses the weak adversary, thereby enhancing the generality of our framework and theory. However, we argue that the strong adversary is more practical in many real-world scenarios, and bringing this stronger notion has further practical impacts beyond blindly generalizing our model.

Consider a scenario where an adversary conducts a membership inference attack on a large scale. We argue that in practice, it is more reasonable to aim for a high *overall membership accuracy* of a set of carefully chosen data points, rather than the individual membership status among each of them. For example, consider the case that we are interested in images sourced from the internet. In this case, it is safe to assume that images within the same webpage are either all included in the model training dataset or none are if we assume that the training data is collected via some reasonable data mining algorithms. In such a case, the ability of an adversary to infer the common membership for a group of data points from a particular webpage becomes desirable as it is likely to enhance the overall MIA accuracy. We believe that this stronger notion of MIA adversary has more practical impacts and reflects the common practice when deploying the membership inference attack, therefore we choose to formulate the unlearning sample inference game with it.

B.3 Proof of Theorem 3.3

We now prove Theorem 3.3. We repeat the statement for convenience.

Theorem B.1. *For any (potentially inefficient) adversary \mathcal{A} , its advantage against the retraining method RETRAIN in an unlearning sample inference game $\mathcal{G} = (\text{RETRAIN}, \mathcal{A}, \mathcal{D}, \mathbb{P}_{\mathcal{D}}, \alpha)$ is zero, i.e., $\text{Adv}(\mathcal{A}, \text{RETRAIN}) = 0$.*

Proof. Firstly, we may partition the collection of all the possible dataset splits \mathcal{S}_{α} by fixing the retain sets $\mathcal{R} \subset \mathcal{D}$. Specifically, denote the collection of dataset splits with the retain set to be \mathcal{R} as $\mathcal{S}_{\alpha}[\mathcal{R}] := \{s \in \mathcal{S}_{\alpha} : s = (\mathcal{R}, \cdot, \cdot)\}$. With the usual convention, when there’s no dataset split corresponds to \mathcal{R} , $\mathcal{S}_{\alpha}[\mathcal{R}] = \emptyset$. Observe that for any $s \in \mathcal{S}_{\alpha}[\mathcal{R}]$, we can pair it up with another dataset split that *swaps* the forget and test sets in s . In other words, for any $s = (\mathcal{R}, \mathcal{F}, \mathcal{T}) \in \mathcal{S}_{\alpha}[\mathcal{R}]$, we see that $(\mathcal{R}, \mathcal{T}, \mathcal{F})$ is also in $\mathcal{S}_{\alpha}[\mathcal{R}]$ since we assume $|\mathcal{F}| = |\mathcal{T}|$, every dataset split will be paired. In addition, since \mathcal{R} is fixed in $\mathcal{S}_{\alpha}[\mathcal{R}]$, we know that $\mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) =: \mathbb{P}_{\mathcal{M}}(\mathcal{R})$ is the same for all $s \in \mathcal{S}_{\alpha}[\mathcal{R}]$ since the unlearning algorithm UL is RETRAIN, i.e., it only depends on \mathcal{R} . With these observations, we can then combine the paired dataset splits within the expectation.

Specifically, for any $s \in \mathcal{S}_{\alpha}[\mathcal{R}]$, let $s' \in \mathcal{S}_{\alpha}[\mathcal{R}]$ to be s ’s pair, i.e., if $s = (\mathcal{R}, \mathcal{F}, \mathcal{T})$ for some \mathcal{F} and \mathcal{T} , then $s' = (\mathcal{R}, \mathcal{T}, \mathcal{F})$. Finally, for a given \mathcal{R} , let’s denote the collection of all such pairs as

$$\mathcal{P}_{\mathcal{R}} := \{\{s, s'\} \subset \mathcal{S}_{\alpha}[\mathcal{R}] : s = (\mathcal{R}, \mathcal{F}, \mathcal{T}), s' = (\mathcal{R}, \mathcal{T}, \mathcal{F}) \text{ for some } \mathcal{F}, \mathcal{T}\},$$

where we use a set rather than an ordered list for the pair s, s' since we do not want to deal with repetitions. Observe that $\mathcal{O}_s(0) = \mathcal{O}_{s'}(1)$ and $\mathcal{O}_s(1) = \mathcal{O}_{s'}(0)$ since the oracles are constructed

984 with respect to the same preference distribution for all data splits. Hence, we have

$$\begin{aligned}
\text{Adv}(\mathcal{A}, \text{RETRAIN}) &= \frac{1}{|\mathcal{S}_\alpha|} \left| \sum_{s \in \mathcal{S}_\alpha} \left(\Pr_{\substack{m \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) \\ \mathcal{O} = \mathcal{O}_s(0)}}(\mathcal{A}^\mathcal{O}(m) = 1) - \Pr_{\substack{m \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) \\ \mathcal{O} = \mathcal{O}_s(1)}}(\mathcal{A}^\mathcal{O}(m) = 1) \right) \right| \\
&= \frac{1}{|\mathcal{S}_\alpha|} \left| \sum_{\mathcal{R} \subset \mathcal{D}} \sum_{\{s, s'\} \in \mathcal{P}_{\mathcal{R}}} \left(\Pr_{\substack{m \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) \\ \mathcal{O} = \mathcal{O}_s(0)}}(\mathcal{A}^\mathcal{O}(m) = 1) - \Pr_{\substack{m \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) \\ \mathcal{O} = \mathcal{O}_s(1)}}(\mathcal{A}^\mathcal{O}(m) = 1) \right. \right. \\
&\quad \left. \left. + \Pr_{\substack{m \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s') \\ \mathcal{O} = \mathcal{O}_{s'}(0)}}(\mathcal{A}^\mathcal{O}(m) = 1) - \Pr_{\substack{m \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s') \\ \mathcal{O} = \mathcal{O}_{s'}(1)}}(\mathcal{A}^\mathcal{O}(m) = 1) \right) \right| \\
&= \frac{1}{|\mathcal{S}_\alpha|} \left| \sum_{\mathcal{R} \subset \mathcal{D}} \sum_{\{s, s'\} \in \mathcal{P}_{\mathcal{R}}} \left(\Pr_{\substack{m \sim \mathbb{P}_{\mathcal{M}}(\mathcal{R}) \\ \mathcal{O} = \mathcal{O}_s(0)}}(\mathcal{A}^\mathcal{O}(m) = 1) - \Pr_{\substack{m \sim \mathbb{P}_{\mathcal{M}}(\mathcal{R}) \\ \mathcal{O} = \mathcal{O}_s(1)}}(\mathcal{A}^\mathcal{O}(m) = 1) \right. \right. \\
&\quad \left. \left. + \Pr_{\substack{m \sim \mathbb{P}_{\mathcal{M}}(\mathcal{R}) \\ \mathcal{O} = \mathcal{O}_{s'}(0)}}(\mathcal{A}^\mathcal{O}(m) = 1) - \Pr_{\substack{m \sim \mathbb{P}_{\mathcal{M}}(\mathcal{R}) \\ \mathcal{O} = \mathcal{O}_{s'}(1)}}(\mathcal{A}^\mathcal{O}(m) = 1) \right) \right| = 0.
\end{aligned}$$

985

□

986 Before we end this section, we remark that the above proof implies that the SWAP test also grounds
987 the advantage to zero:

988 **Remark B.2.** Consider a pair of swapped splits s and s' . Observe that for RETRAIN,
989 $\mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) = \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s') =: \mathbb{P}_{\mathcal{M}}(\mathcal{R})$ since this probability only depends on \mathcal{R} ,
990 which is the same for s and s' . With $\mathcal{O}_s(0) = \mathcal{O}_{s'}(1)$ and $\mathcal{O}_s(1) = \mathcal{O}_{s'}(0)$, we have

$$\begin{aligned}
\overline{\text{Adv}}_{\{s, s'\}}(\mathcal{A}, \text{RETRAIN}) &= \frac{1}{2} \left| \Pr_{m \sim \mathbb{P}_{\mathcal{M}}(\mathcal{R})}(\mathcal{A}^{\mathcal{O}_s(0)}(m) = 1) - \Pr_{m \sim \mathbb{P}_{\mathcal{M}}(\mathcal{R})}(\mathcal{A}^{\mathcal{O}_s(1)}(m) = 1) \right. \\
&\quad \left. + \Pr_{m \sim \mathbb{P}_{\mathcal{M}}(\mathcal{R})}(\mathcal{A}^{\mathcal{O}_{s'}(1)}(m) = 1) - \Pr_{m \sim \mathbb{P}_{\mathcal{M}}(\mathcal{R})}(\mathcal{A}^{\mathcal{O}_{s'}(0)}(m) = 1) \right| = 0.
\end{aligned}$$

991 B.4 Proof of Theorem 3.5

992 We prove Theorem 3.5 in this section. Before this, we formally introduce the notion of *certified*
993 *removal*.

994 **Definition B.3** (Certified Removal [Guo et al., 2020]). For a fixed dataset \mathcal{D} , let LR and UL be a
995 learning and an unlearning algorithm respectively, and denote $\mathcal{H} := \text{im}(\text{LR}) \cup \text{im}(\text{UL})$ ⁵ to be the
996 hypothesis class containing all possible models that can be produced by LR and UL. Then, for any
997 $\epsilon, \delta > 0$, the unlearning algorithm UL is said to be (ϵ, δ) -certified removal if for any $\mathcal{W} \subset \mathcal{H}$ and for
998 any disjoint $\mathcal{R}, \mathcal{F} \subseteq \mathcal{D}$ (do not need to satisfy restriction (a) and (b)),

$$\begin{aligned}
\Pr(\text{UL}(\text{LR}(\mathcal{R} \cup \mathcal{F}), \mathcal{F}) \in \mathcal{W}) &\leq e^\epsilon \Pr(\text{RETRAIN}(\text{LR}(\mathcal{R} \cup \mathcal{F}), \mathcal{F}) \in \mathcal{W}) + \delta; \\
\Pr(\text{RETRAIN}(\text{LR}(\mathcal{R} \cup \mathcal{F}), \mathcal{F}) \in \mathcal{W}) &\leq e^\epsilon \Pr(\text{UL}(\text{LR}(\mathcal{R} \cup \mathcal{F}), \mathcal{F}) \in \mathcal{W}) + \delta.
\end{aligned}$$

999 We note that as $\text{RETRAIN}(\text{LR}(\mathcal{R} \cup \mathcal{F}), \mathcal{F}) = \text{LR}(\mathcal{R})$, the above can be simplified to

$$\begin{aligned}
\Pr(\text{UL}(\text{LR}(\mathcal{R} \cup \mathcal{F}), \mathcal{F}) \in \mathcal{W}) &\leq e^\epsilon \Pr(\text{LR}(\mathcal{R}) \in \mathcal{W}) + \delta \\
\Pr(\text{LR}(\mathcal{R}) \in \mathcal{W}) &\leq e^\epsilon \Pr(\text{UL}(\text{LR}(\mathcal{R} \cup \mathcal{F}), \mathcal{F}) \in \mathcal{W}) + \delta.
\end{aligned}$$

1000 Now, we restate Theorem 3.5 for convenience.

⁵Here, $\text{im}(\cdot)$ denote the image of a function.

1001 **Theorem B.4.** Given an (ϵ, δ) -certified removal unlearning algorithm UL with some $\epsilon, \delta > 0$, for
 1002 any (potentially inefficient) adversary \mathcal{A} against UL in an unlearning sample inference game \mathcal{G} , we
 1003 have

$$\text{Adv}(\mathcal{A}, \text{UL}) \leq 2 \cdot \left(1 - \frac{2 - 2\delta}{e^\epsilon + 1}\right).$$

1004 *Proof.* We start by considering an attack as differentiating between the following two hypotheses:
 1005 the unlearning and the retraining. In particular, given a specific dataset split $s = (\mathcal{R}, \mathcal{F}, \mathcal{T}) \in \mathcal{S}_\alpha$ and
 1006 an model m , consider

$$H_1: m = \text{UL}(\text{LR}(\mathcal{R} \cup \mathcal{F}), \mathcal{F}), \text{ and } H_2: m = \text{LR}(\mathcal{R}) = \text{RETRAIN}(\text{LR}(\mathcal{R} \cup \mathcal{F}), \mathcal{F}).$$

1007 Alternatively, by writing the distribution of the unlearned models and the retrained models as
 1008 $\mathbb{P}_{\mathcal{M}}(\text{UL}, s)$ and $\mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s)$, respectively, we may instead write

$$H_1: m \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s), \text{ and } H_2: m \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s).$$

1009 It turns out that by looking at the *type-I error* α and *type-II error* β , we can control the advantage of
 1010 the adversary in this game easily. Firstly, denote the model produced under H_1 as m_1 , then under
 1011 H_1 , the accuracy of the adversary is $\Pr_{m_1 \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s)}(\mathcal{A}(m_1) = 1) = 1 - \alpha$. Similarly, by denoting
 1012 the model produced under H_2 as m_2 , we have $\Pr_{m_2 \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s)}(\mathcal{A}(m_2) = 1) = \beta$. Therefore,
 1013 for this specific dataset split s , let's define the advantage of this adversary \mathcal{A} for this attack as⁶

$$\widehat{\text{Adv}}_s(\mathcal{A}) := |1 - \alpha - \beta|.$$

1014 The upshot is that since UL is an (ϵ, δ) -certified removal unlearning algorithm (Definition B.3), it is
 1015 possible to control α and β , hence $\widehat{\text{Adv}}_s(\mathcal{A})$. To achieve this, since from the definition of certified
 1016 removal, we're dealing with sub-collections of models, it helps to write α and β differently.

1017 Let $\mathcal{H} := \text{supp}(\mathbb{P}_{\mathcal{M}}(\text{UL}, s)) \cup \text{supp}(\mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s))$ be the collection of all possible models,
 1018 and denote $\mathcal{B} \subseteq \mathcal{H}$ to be the collection of models that the adversary \mathcal{A} accepts, and \mathcal{B}^c to denote its
 1019 complement, i.e., the collection of models that the adversary \mathcal{A} rejects. We can then re-write the
 1020 type-I and type-II errors as

- 1021 • Type-I error α : probability of rejecting H_1 when H_1 is true, i.e., $\alpha = \Pr(m_1 \in \mathcal{B}^c \mid s) =$
 1022 $1 - \Pr(m_1 \in \mathcal{B} \mid s)$.
- 1023 • Type-II error β : probability of accepting H_2 when H_2 is false, i.e., $\beta = \Pr(m_2 \in \mathcal{B} \mid s)$.

1024 With this interpretation and the fact that UL is (ϵ, δ) -certified removal, we know that

- 1025 • $\Pr(m_1 \in \mathcal{B} \mid s) \leq e^\epsilon \Pr(m_2 \in \mathcal{B} \mid s) + \delta$, and
- 1026 • $\Pr(m_2 \in \mathcal{B}^c \mid s) \leq e^\epsilon \Pr(m_1 \in \mathcal{B}^c \mid s) + \delta$.

1027 Combining the above, we have $1 - \alpha \leq e^\epsilon \beta + \delta$ and $1 - \beta \leq e^\epsilon \alpha + \delta$. Hence,

$$\beta \geq \max\{0, 1 - \delta - e^\epsilon \alpha, e^{-\epsilon}(1 - \delta - \alpha)\}.$$

1028 We then seek to get the minimum of $\alpha + \beta$, we have

$$\alpha + \beta \geq \max\{\alpha, 1 - \delta - e^\epsilon \alpha + \alpha, e^{-\epsilon}(1 - \delta - \alpha) + \alpha\}.$$

1029 To get a lower bound, consider the minimum among the last two, i.e., consider solving α when
 1030 $1 - \delta - e^\epsilon \alpha + \alpha = e^{-\epsilon}(1 - \delta - \alpha) + \alpha$, leading to

$$(e^{-\epsilon} - e^\epsilon)\alpha = e^{-\epsilon}(1 - \delta) - (1 - \delta) = (e^{-\epsilon} - 1)(1 - \delta) \implies \alpha = \frac{(e^{-\epsilon} - 1)(1 - \delta)}{e^{-\epsilon} - e^\epsilon}.$$

1031 Hence, we have

$$\alpha + \beta \geq 1 - \delta + \alpha(1 - e^\epsilon) = 1 - \delta + \frac{(e^{-\epsilon} - 1)(1 - \delta)}{e^{-\epsilon} - e^\epsilon}(1 - e^\epsilon) = (1 - \delta) \frac{2e^{-\epsilon} - 2}{e^{-\epsilon} - e^\epsilon} = (1 - \delta) \frac{2 - 2e^\epsilon}{1 - e^{2\epsilon}},$$

⁶Note that this is different from the advantage we defined before since the attack is different, hence we use a different notation.

1032 with the elementary identity $1 - e^{2\epsilon} = (1 + e^\epsilon)(1 - e^\epsilon)$, we finally get

$$\alpha + \beta \geq \frac{2 - 2\delta}{e^\epsilon + 1}.$$

1033 On the other hand, considering the “dual attack” that predicts the opposite as the original attack,
 1034 that is, we flip \mathcal{B} and \mathcal{B}^c . In this case, the type-I error and the type-II error become α and $1 - \beta$,
 1035 respectively. Following the same procedures, we’ll have $\alpha + \beta \leq 2 - \frac{2-2\delta}{e^\epsilon+1}$.

1036 Note that the definition of (ϵ, δ) -certified removal is independent of the dataset split s , hence, the
 1037 above derivation works for all s . In particular, the advantage of any adversary differentiating H_1 and
 1038 H_2 for any s is upper bounded by

$$\widehat{\text{Adv}}_s(\mathcal{A}) = |1 - \alpha - \beta| \leq 1 - \frac{2 - 2\delta}{e^\epsilon + 1} =: \tau,$$

1039 where we denote τ to be the upper bound of advantage. This means that **any** adversaries trying to
 1040 differentiate between retrain models and certified unlearned models are upper bounded in terms of its
 1041 advantage, and an explicit upper bound is given by τ .

1042 We now show a reduction from the unlearning sample inference game to the above. Firstly, we
 1043 construct two attacks based on the adversary \mathcal{A} in the unlearning sample inference game, which tries
 1044 to differentiate between the data point x is sampled from the forget set \mathcal{F} or the test set \mathcal{T} . This can
 1045 be formulated through the following hypotheses testing:

$$H_3: x \in \mathcal{F}, \text{ and } H_4: x \in \mathcal{T}.$$

1046 In this viewpoint, the unlearning sample inference game can be thought of as deciding between
 1047 H_3 and H_4 . Since the upper bound we get for differentiating between H_1 and H_2 holds for any
 1048 efficient adversaries, therefore, we can construct an attack for deciding between H_1 and H_2 using
 1049 adversaries for deciding between H_3 and H_4 . This allows us to upper bound the advantage for the
 1050 latter adversaries.

1051 Given any adversaries \mathcal{A} for differentiating H_3 and H_4 , i.e., any adversaries in the unlearning sample
 1052 inference game, we start by constructing our first adversary \mathcal{A}_1 for differentiating H_1 and H_2 as
 1053 follows:

- 1054 • In the left world (H_1), feed the certified unlearned model to \mathcal{A} ; in the right world H_2 , feed
 1055 the retrained model to \mathcal{A} .
- 1056 • We create a random oracle $\mathcal{O}_s(0)$ for \mathcal{A} , i.e., we let the adversary \mathcal{A} decide on \mathcal{F} . We then
 1057 let \mathcal{A}_1 output as \mathcal{A} .

1058 We note that \mathcal{A} is deciding on \mathcal{F} , the advantage of \mathcal{A}_1 is

$$\widehat{\text{Adv}}_s(\mathcal{A}_1) = \left| \Pr_{\substack{m_1 \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s) \\ \mathcal{O} = \mathcal{O}_s(0)}} (\mathcal{A}^\mathcal{O}(m_1) = 1) - \Pr_{\substack{m_2 \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) \\ \mathcal{O} = \mathcal{O}_s(0)}} (\mathcal{A}^\mathcal{O}(m_2) = 1) \right|.$$

1059 We can also induce the average of the advantage over all dataset splits is upper bounded by the
 1060 maximal advantage taken over all dataset splits:

$$\begin{aligned} & \frac{1}{|\mathcal{S}_\alpha|} \left| \sum_{s \in \mathcal{S}_\alpha} \Pr_{\substack{m_1 \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s) \\ \mathcal{O} = \mathcal{O}_s(0)}} (\mathcal{A}^\mathcal{O}(m_1) = 1) - \sum_{s \in \mathcal{S}_\alpha} \Pr_{\substack{m_2 \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) \\ \mathcal{O} = \mathcal{O}_s(0)}} (\mathcal{A}^\mathcal{O}(m_2) = 1) \right| \\ & \leq \frac{1}{|\mathcal{S}_\alpha|} \sum_{s \in \mathcal{S}_\alpha} \left| \Pr_{\substack{m_1 \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s) \\ \mathcal{O} = \mathcal{O}_s(0)}} (\mathcal{A}^\mathcal{O}(m_1) = 1) - \Pr_{\substack{m_2 \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) \\ \mathcal{O} = \mathcal{O}_s(0)}} (\mathcal{A}^\mathcal{O}(m_2) = 1) \right| \leq \max_{s \in \mathcal{S}_\alpha} \widehat{\text{Adv}}_s(\mathcal{A}_1). \end{aligned}$$

1061 Similarly, we can construct a second adversary \mathcal{A}_2 for differentiating H_1 and H_2 as follows:

- 1062 • In the left world (H_1), feed the certified unlearned model to \mathcal{A} . In the right world (H_2), feed
 1063 retrained model to \mathcal{A} .

1064 • We create a random oracle $O_s(1)$ for \mathcal{A} , i.e., we let the adversary \mathcal{A} decide on \mathcal{T} . We then
 1065 let \mathcal{A}_2 outputs as the \mathcal{A} .

1066 Since \mathcal{A} is deciding on \mathcal{T} , the advantage of \mathcal{A}_2 is

$$\widehat{\text{Adv}}_s(\mathcal{A}_2) = \left| \Pr_{\substack{m_1 \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s) \\ \mathcal{O} = \mathcal{O}_s(1)}} (\mathbb{1}(\mathcal{A}^{\mathcal{O}}(m_1) = 1)) - \Pr_{\substack{m_2 \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) \\ \mathcal{O} = \mathcal{O}_s(1)}} (\mathcal{A}^{\mathcal{O}}(m_2) = 1) \right|.$$

1067 Similar to the previous calculation for \mathcal{A}_1 , the average of the advantage is also upper bounded by the
 1068 maximal advantage,

$$\begin{aligned} & \frac{1}{|\mathcal{S}_{\alpha}|} \left| \sum_{s \in \mathcal{S}_{\alpha}} \Pr_{\substack{m_1 \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s) \\ \mathcal{O} = \mathcal{O}_s(1)}} (\mathcal{A}^{\mathcal{O}}(m_1) = 1) - \sum_{s \in \mathcal{S}_{\alpha}} \Pr_{\substack{m_2 \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) \\ \mathcal{O} = \mathcal{O}_s(1)}} (\mathcal{A}^{\mathcal{O}}(m_2) = 1) \right| \\ & \leq \frac{1}{|\mathcal{S}_{\alpha}|} \sum_{s \in \mathcal{S}_{\alpha}} \left| \Pr_{\substack{m_1 \sim \mathbb{P}_{\mathcal{M}}(\mathcal{M}_1 | \mathcal{S}_{\alpha}) \\ \mathcal{O} = \mathcal{O}_s(1)}} (\mathcal{A}^{\mathcal{O}}(m_1) = 1) - \Pr_{\substack{m_2 \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) \\ \mathcal{O} = \mathcal{O}_s(1)}} (\mathcal{A}^{\mathcal{O}}(m_2) = 1) \right| \leq \max_{s \in \mathcal{S}_{\alpha}} \widehat{\text{Adv}}_s(\mathcal{A}_2). \end{aligned}$$

1069 Given the above calculation, we can now bound the advantage of \mathcal{A} . Firstly, let UL be any certified
 1070 unlearning method. Then the advantage $\text{Adv}(\mathcal{A}, \text{UL})$ of \mathcal{A} in the unlearning sample inference game
 1071 (i.e., differentiating between H_3 and H_4) against UL is

$$\frac{1}{|\mathcal{S}_{\alpha}|} \left| \sum_{s \in \mathcal{S}_{\alpha}} \Pr_{\substack{m_1 \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s) \\ \mathcal{O} = \mathcal{O}_s(0)}} (\mathcal{A}^{\mathcal{O}}(m_1) = 1) - \sum_{s \in \mathcal{S}_{\alpha}} \Pr_{\substack{m_1 \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s) \\ \mathcal{O} = \mathcal{O}_s(1)}} (\mathcal{A}^{\mathcal{O}}(m_1) = 1) \right|.$$

1072 On the other hand, the advantage $\text{Adv}(\mathcal{A}, \text{RETRAIN})$ of \mathcal{A} against the retraining method RETRAIN
 1073 can be written as

$$\frac{1}{|\mathcal{S}_{\alpha}|} \left| \sum_{s \in \mathcal{S}_{\alpha}} \Pr_{\substack{m_2 \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) \\ \mathcal{O} = \mathcal{O}_s(0)}} (\mathcal{A}^{\mathcal{O}}(m_2) = 1) - \sum_{s \in \mathcal{S}_{\alpha}} \Pr_{\substack{m_2 \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) \\ \mathcal{O} = \mathcal{O}_s(1)}} (\mathcal{A}^{\mathcal{O}}(m_2) = 1) \right|,$$

1074 which is indeed 0 from Theorem 3.3. Combine this with the calculations above, from the reverse
 1075 triangle inequality,

$$\begin{aligned} \text{Adv}(\mathcal{A}, \text{UL}) &= |\text{Adv}(\mathcal{A}, \text{UL}) - \text{Adv}(\mathcal{A}, \text{RETRAIN})| \\ &\leq \frac{1}{|\mathcal{S}_{\alpha}|} \left| \sum_{s \in \mathcal{S}_{\alpha}} \Pr_{\substack{m_2 \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) \\ \mathcal{O} = \mathcal{O}_s(0)}} (\mathcal{A}^{\mathcal{O}}(m_2) = 1) - \sum_{s \in \mathcal{S}_{\alpha}} \Pr_{\substack{m_2 \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) \\ \mathcal{O} = \mathcal{O}_s(1)}} (\mathcal{A}^{\mathcal{O}}(m_2) = 1) \right. \\ &\quad \left. - \sum_{s \in \mathcal{S}_{\alpha}} \Pr_{\substack{m_1 \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s) \\ \mathcal{O} = \mathcal{O}_s(0)}} (\mathcal{A}^{\mathcal{O}}(m_1) = 1) + \sum_{s \in \mathcal{S}_{\alpha}} \Pr_{\substack{m_1 \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s) \\ \mathcal{O} = \mathcal{O}_s(1)}} (\mathcal{A}^{\mathcal{O}}(m_1) = 1) \right| \\ &\leq \frac{1}{|\mathcal{S}_{\alpha}|} \sum_{s \in \mathcal{S}_{\alpha}} \left| \Pr_{\substack{m_1 \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s) \\ \mathcal{O} = \mathcal{O}_s(0)}} (\mathcal{A}^{\mathcal{O}}(m_1) = 1) - \Pr_{\substack{m_2 \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) \\ \mathcal{O} = \mathcal{O}_s(0)}} (\mathcal{A}^{\mathcal{O}}(m_2) = 1) \right| \\ &\quad + \frac{1}{|\mathcal{S}_{\alpha}|} \sum_{s \in \mathcal{S}_{\alpha}} \left| \Pr_{\substack{m_1 \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s) \\ \mathcal{O} = \mathcal{O}_s(1)}} (\mathcal{A}^{\mathcal{O}}(m_1) = 1) - \Pr_{\substack{m_2 \sim \mathbb{P}_{\mathcal{M}}(\text{RETRAIN}, s) \\ \mathcal{O} = \mathcal{O}_s(1)}} (\mathcal{A}^{\mathcal{O}}(m_2) = 1) \right| \\ &\leq \max_{s \in \mathcal{S}_{\alpha}} \widehat{\text{Adv}}_s(\mathcal{A}_1) + \max_{s \in \mathcal{S}_{\alpha}} \widehat{\text{Adv}}_s(\mathcal{A}_2) \\ &\leq 2\tau, \end{aligned}$$

1076 i.e., the advantage of any adversary against any certified unlearning method is bounded by 2τ . \square

1077 B.5 Proof of Proposition 3.7

1078 We now prove Proposition 3.7. We repeat the statement for convenience.

1079 **Proposition B.5.** *For any two dataset splits $s_1, s_2 \in \mathcal{S}_\alpha$ satisfying non-degeneracy assumption, i.e.,*
 1080 *both $\mathbb{P}_{\mathcal{D}}|_{\mathcal{F}_i}(\mathcal{F}_1 \cap \mathcal{F}_2)$ and $\mathbb{P}_{\mathcal{D}}|_{\mathcal{T}_i}(\mathcal{T}_1 \cap \mathcal{T}_2)$ do not vanish polynomially faster in $|\mathcal{D}|$, then there*
 1081 *exists a deterministic and efficient adversary \mathcal{A} such that $\overline{\text{Adv}}_{\{s_1, s_2\}}(\mathcal{A}, \text{UL}) = 1$ for any unlearning*
 1082 *method UL. In particular, $\overline{\text{Adv}}_{\{s_1, s_2\}}(\mathcal{A}, \text{RETRAIN}) = 1$.*

1083 *Proof.* Consider any unlearning method UL, and design a random oracle \mathcal{O}_{s_i} based on the split s_i
 1084 for $i = 1, 2$ and a sensitivity distribution $\mathbb{P}_{\mathcal{D}}$ (which for simplicity, assume to have full support across
 1085 \mathcal{D}), we see that

$$\begin{aligned} \overline{\text{Adv}}_{\{s_1, s_2\}}(\mathcal{A}, \text{UL}) = & \frac{1}{2} \left| \Pr_{\substack{m \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s_1) \\ \mathcal{O} = \mathcal{O}_{s_1}(0)}}(\mathcal{A}^{\mathcal{O}}(m) = 1) - \Pr_{\substack{m \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s_1) \\ \mathcal{O} = \mathcal{O}_{s_1}(1)}}(\mathcal{A}^{\mathcal{O}}(m) = 1) \right. \\ & \left. + \Pr_{\substack{m \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s_2) \\ \mathcal{O} = \mathcal{O}_{s_2}(0)}}(\mathcal{A}^{\mathcal{O}}(m) = 1) - \Pr_{\substack{m \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s_2) \\ \mathcal{O} = \mathcal{O}_{s_2}(1)}}(\mathcal{A}^{\mathcal{O}}(m) = 1) \right|. \end{aligned}$$

1086 Consider a hard-coded adversary \mathcal{A} which has a look-up table T , defined as

$$T(x) = \begin{cases} 1, & \text{if } x \in \mathcal{T}_1 \cap \mathcal{T}_2; \\ 0, & \text{if } x \in \mathcal{F}_1 \cap \mathcal{F}_2; \\ \perp, & \text{otherwise,} \end{cases}$$

1087 where we use \perp to denote an undefined output. Then, \mathcal{A} predicts the bit b used by the oracle as follows: We see that Algorithm 1 with a hard-coded look-up table T has several properties:

Algorithm 1: Dummy adversary \mathcal{A} against a random 2-sets evaluation

Data: An unlearned model m , a random oracle \mathcal{O}

Result: A one bit prediction b

```

b  $\leftarrow \perp$ 
while  $b = \perp$  do
   $x \sim \mathcal{O}$ 
   $b \leftarrow T(x)$ 
return  $b$ 

```

1088

1089 • Since it neglects m entirely,

$$\Pr_{\substack{m \sim \mathbb{P}_{\mathcal{M}}(\text{UL}, s_i) \\ \mathcal{O} = \mathcal{O}_{s_i}(b)}}(\mathcal{A}^{\mathcal{O}}(m) = 1) = \Pr_{\mathcal{O} = \mathcal{O}_{s_i}(b)}(\mathcal{A}^{\mathcal{O}} = 1)$$

1090 • Under the non-degeneracy assumptions, \mathcal{A} will terminate in polynomial time in $|\mathcal{D}|$. This
 1091 is because the expected terminating time is inversely proportional to $\mathbb{P}_{\mathcal{D}}|_{\mathcal{F}_i}(\mathcal{F}_1 \cap \mathcal{F}_2)$ and
 1092 $\mathbb{P}_{\mathcal{D}}|_{\mathcal{T}_i}(\mathcal{T}_1 \cap \mathcal{T}_2)$, hence if these two probabilities does not vanish polynomially faster in $|\mathcal{D}|$
 1093 (i.e., the non-degeneracy assumption), then it'll terminate in polynomial time in $|\mathcal{D}|$.

1094 • Whenever \mathcal{A} terminates and outputs an answer, it will be correct, i.e.,

$$\overline{\text{Adv}}_{\{s_1, s_2\}}(\mathcal{A}, \text{UL}) = 1.$$

1095 Since the above argument works for every UL, hence even for the retraining method RETRAIN, we
 1096 will have $\overline{\text{Adv}}_{\{s_1, s_2\}}(\mathcal{A}, \text{RETRAIN}) = 1$. Intuitively, such a pathological case can happen since there
 1097 exists some \mathcal{A} which interpolates the “correct answer” for a few splits. Though adversaries may not
 1098 have access to specific dataset splits, learning-based attacks could still undesirably learn towards
 1099 this scenario if evaluated only on a few splits. Thus, we should penalize hard-coded adversaries in
 1100 evaluation. \square

1101 B.6 Discussion on A Naive Strategy Offsetting MIA Accuracy

1102 In this section, we consider a toy example to illustrate the difference between the proposed advantage-
1103 based metric and a naive strategy that simply offsets the MIA accuracy for RETRAIN to zero.

1104 Let us assume there are 6 data points $\{A, B, C, D, E, F\}$, and we equally split them into the forget
1105 set \mathcal{F} and the test set \mathcal{T} . Running MIA against RETRAIN on a retrained model m^* trained on the
1106 retain set \mathcal{R} independent of the above 6 data points, the MIA predicts the probability that each data
1107 point belongs to the forget set is:

$$\begin{aligned} \text{MIA}(m^*, A) &= 0.7, & \text{MIA}(m^*, B) &= 0.4, & \text{MIA}(m^*, C) &= 0.3, \\ \text{MIA}(m^*, D) &= 0.1, & \text{MIA}(m^*, E) &= 0.6, & \text{MIA}(m^*, F) &= 0.8. \end{aligned}$$

1108 Assume that the MIA adversary \mathcal{A} chooses the cutoff of predicting a data point belonging to the forget
1109 set to be ≥ 0.5 , then the prediction will be

$$\mathcal{A}(A) = 1, \quad \mathcal{A}(B) = 0, \quad \mathcal{A}(C) = 0, \quad \mathcal{A}(D) = 0, \quad \mathcal{A}(E) = 1, \quad \mathcal{A}(F) = 1,$$

1110 where 1 refers to the forget set while 0 refers to the test set. Since the retrained model will be the
1111 same regardless of the split of the forget set and the test set, the MIA prediction will be the same
1112 regardless of the split as well.

1113 Assuming that we have two imperfect unlearning algorithms, UL_1 and UL_2 . For simplicity, we could
1114 assume running MIA on the unlearned model m_1 by UL_1 will increase the predicted probability on
1115 the forget set by 0.1, while that (denoted as m_2) by UL_2 will increase it by 0.2. For example, if we
1116 set $\mathcal{F} = \{A, B, C\}$ while $\mathcal{T} = \{D, E, F\}$, then the MIA on UL_1 will predict the probability as

$$\begin{aligned} \text{MIA}(m_1, A) &= 0.8, & \text{MIA}(m_1, B) &= 0.5, & \text{MIA}(m_1, C) &= 0.4, \\ \text{MIA}(m_1, D) &= 0.1, & \text{MIA}(m_1, E) &= 0.6, & \text{MIA}(m_1, F) &= 0.8, \end{aligned}$$

1117 while the MIA on UL_2 will predict the probability as

$$\begin{aligned} \text{MIA}(m_2, A) &= 0.9, & \text{MIA}(m_2, B) &= 0.6, & \text{MIA}(m_2, C) &= 0.5, \\ \text{MIA}(m_2, D) &= 0.1, & \text{MIA}(m_2, E) &= 0.6, & \text{MIA}(m_2, F) &= 0.8, \end{aligned}$$

1118 Intuitively, UL_2 is worse than UL_1 in terms of unlearning quality. In this simplified setup, we claim
1119 the following.

1120 **Claim B.6.** *The advantage calculated by one SWAP test over RETRAIN is always 0, while the*
1121 *advantage calculated by one SWAP test over UL_1 and UL_2 are respectively $1/6$ and $1/3$, all*
1122 *regardless of the exact split of the data points. Hence, the Unlearning Quality for RETRAIN, UL_1 , and*
1123 *UL_2 are respectively 1, $5/6$, and $2/3$, faithfully reflecting the unlearning algorithms' performances.*

1124 *On the other hand, the "offset MIA accuracy" is dependent on the split of the data. Specifically, when*
1125 *we assign $\{D, E, F\}$ as the forget set and $\{A, B, C\}$ as the test set, the MIA accuracies for all three*
1126 *methods are the same, making the "offset MIAs" all equal to 0.5, failing to capture the unlearning*
1127 *quality.*

1128 *Proof.* Given a split s , denote the predicted MIA result as $\hat{Y}_i^s \in \{0, 1\}$, and the actual membership as
1129 $Y_i^s = \mathbb{1}_{i \in \mathcal{F}}$ for $i \in \{A, B, \dots, F\}$. Then, consider a simple adversary \mathcal{A} : after getting the predicted
1130 MIA probability, i.e., $\Pr(\hat{Y}_i^s = 1)$, we update \hat{Y}_i^s as 1 if $\Pr(\hat{Y}_i^s = 1) \geq 1/2$, and 0 otherwise. Then,
1131 the advantage of a particular split s is $\Pr_m(\mathcal{A}^{\mathcal{O}_s(0)}(m) = 1) - \Pr_m(\mathcal{A}^{\mathcal{O}_s(1)}(m) = 1) = \Pr_i(\hat{Y}_i^s =$
1132 $1 \mid Y_i^s = 0) - \Pr_i(\hat{Y}_i^s = 1 \mid Y_i^s = 1)$ for $i \in \{A, B, \dots, F\}$. These probabilities are essentially an
1133 average of indicator variables: for example, $\Pr_i(\hat{Y}_i^s = 1 \mid Y_i^s = 0) = \sum_{i: Y_i^s = 0} \mathbb{1}_{\hat{Y}_i^s = 1} / |\{i: Y_i^s =$
1134 $0\}|$, and since we're considering equal-size splits, $|\{i: Y_i^s = 0\}| = |\{i: Y_i^s = 1\}| = 6/2 = 3$ in our
1135 example.

1136 To see how the calculation works out, we note that for a SWAP split (s, s') , each i appears in the
1137 forget set and the test set exactly once when calculating the SWAP advantage, i.e., $Y_i^s = 1$ and
1138 $Y_i^{s'} = 0$ or $Y_i^s = 0$ and $Y_i^{s'} = 1$. Furthermore, suppose \hat{Y}_i^s and $\hat{Y}_i^{s'}$ are the same, e.g., when
1139 considering RETRAIN, then the indicators $\mathbb{1}_{\hat{Y}_i^s = 1}$ and $\mathbb{1}_{\hat{Y}_i^{s'} = 1}$ will be the same and appear in pair
1140 (specifically, with opposite sign, one in Adv_s and another in $\text{Adv}_{s'}$), and hence cancel each other out.
1141 This is why the advantage is 0 for RETRAIN. This pairing of indicators for i under splits s and s'

happens for imperfect unlearning algorithms, but the indicator might change: \hat{Y}_i^s can swap from 0 to 1 due to imperfect unlearning. If this happens, i contributes 1/3 to the denominator of the SWAP advantage formula, hence 1/6 in total (divided by 2 at the end). With this observation, we see that for UL_1 , only B 's prediction will be flipped from 0 to 1, hence UL_1 's advantage is 1/6 in the SWAP test. The same argument applies for UL_2 where both B and C 's predictions are flipped, hence the advantage is $2/6 = 1/3$. \square

C Omitted Details From Section 4

C.1 Computational Resource and Complexity

We conduct our experiment on Intel(R) Xeon(R) Gold 6338 CPU @ 2.00GHz with 4 A40 NVIDIA GPUs. It takes approximately 6 days to reproduce the experiment of standard deviation comparison between SWAP test and random dataset splitting. It takes approximately 1 day to reproduce the experiment of dataset size and random seeds. Furthermore, it takes approximately 4 days to reproduce the experiment of differential private testing.

C.2 Details of Training

For target model training without differential privacy (DP) guarantees, we consider using the ResNet-20 [He et al., 2016] as our target model and train it with Stochastic Gradient Descent (SGD) [Ruder, 2016] optimizer with a MultiStepLR learning rate scheduler with milestones [100, 150] and an initial learning rate of 0.1, momentum 0.9, weight decay 10^{-5} . Moreover, we train the model with 200 epoch, which we empirically observe that this guarantees a convergence. For a given dataset split, we average 3 models to approximate the randomness induced in training and unlearning procedures.

For training DP models, we use DP-SGD [Abadi et al., 2016] to provide DP guarantees. Specifically, we adopt the OPACUS implementation [Yousefpour et al., 2021] and use ResNet-18 [He et al., 2016] as our target model. The model is trained with the RMSProp optimizer using a learning rate 0.01 and of 20 epochs. This ensures convergence as we empirically observe that 20 epochs suffice to yield a comparable model accuracy. Considering the dataset size, we use $\delta = 10^{-5}$ and tune the max gradient norm individually.

C.3 IC Score and MIA Score

One of the two metrics we choose to compare against is the Interclass Confusion (IC) Test [Goel et al., 2023]. In brief, the IC test ‘‘confuses’’ a selected set of two classes by switching their labels. This is implemented by picking two classes and randomly selecting half of the data from each class for confusion. Then the IC test proceeds to train the corresponding target models on the new datasets and perform unlearning on the selected set using the unlearning algorithm being tested, and finally measures the inter-class error of the unlearned models on the selected set, which we called the *memorization score* γ . Similar to the advantage, the memorization score is between $[0, 1]$, and the lower, the better since ideally, the unlearned model should have no memorization of the confusion. Given this, to compare the IC test with the Unlearning Quality \mathcal{Q} , we consider $1 - \gamma$, and refer to this new score as the *IC score*.

On the other hand, the MIA AUC is a popular MIA-based metric to measure the performance of the unlearning. It measures how MIA performs by calculating the AUC (Area Under Curve) of MIA on the union of the test set and the forget set. We note that AUC is a widely used evaluation metric in terms of classification models since compared to directly measuring the accuracy, AUC tends to measure how well the model can discriminate against each class. Finally, as defined in Section 4, we let the *MIA score* to be $1 - \text{MIA AUC}$ to have a fair comparison.

Model Accuracy versus DP Budgets. We also report the classification accuracy of the original model trained with various DP budgets in Table 3. As can be seen, the classification accuracy increases as the ϵ is relaxed to a larger value, showing the inherent trade-off between DP and utility. For experiments about Unlearning Quality in Table 1 and MIA score in Table 2(B), the original models are shared and thus have the same results (Table 3(B)). We note that measuring the IC scores requires dataset modifications, so the model accuracy in the experiments of IC score (Table 3(A)) differs slightly from those in experiments of Unlearning Quality and MIA score (Table 3(B)).

Table 3: Model accuracy *versus* DP Budgets. See Table 1 for more context.

(A) Results for experiments in Table 2(A)					(B) Results for experiments in Tables 1 and 2(B)				
ϵ	50	150	600	∞	ϵ	50	150	600	∞
Accuracy	0.442*	0.506*	0.540*	0.639*	Accuracy	0.485*	0.520*	0.571*	0.660*

Remark C.1. We would like to clarify the performance difference compared to common literature, which can be attributed to the dataset split. The original dataset is split evenly into the target and shadow datasets for the purpose of implementing MIA. Within the target dataset, further partitioning is performed to create the retain, forget, and test sets. As a result, only about 30% of the full dataset remains available for training the model, significantly reducing the effective training data. Note that the data split is necessary for our experiments so we cannot get significantly more training data.

We experimented with training on the full dataset and applied data augmentation while keeping all other configurations unchanged. With the full dataset, the model achieved an accuracy of 85.34%. After incorporating data augmentation, the accuracy further improved to 91.13%, aligning with the past literature. This simple ablation study validates that the performance difference mainly comes from the difference in data size and the omission of data augmentation.

We select large ϵ in our differential privacy experiments due to the significant drop in accuracy observed when ϵ is small, which stems from the same dataset size limitation. We include an additional experiment with smaller ϵ in the linear setting, as presented at the end of Appendix C.4.

C.4 Additional Experiments

In this section, we provide additional ablation experiments on our proposed Unlearning Quality metric by considering varying various parameters and settings.

Unlearning Quality Versus Dataset Size. We provide additional experiments under different dataset sizes. The experiment is conducted with the ResNet20 model architecture, CIFAR10 dataset, and with unlearning portion parameter $\alpha = 0.1$. The results are shown in Table 4. Observe that the *relative ranking* of different unlearning methods stays mostly consistent across different dataset sizes. This suggests that model maintainers can potentially extrapolate the evaluation result of the Unlearning Quality from smaller-scale to larger-scale experiments. This scalability enhances the efficiency of our metric, facilitating the selection of the most effective unlearning method without the necessity of extensive resource expenditure.

Table 4: Unlearning Quality *versus* dataset size η (in percentage). The relative ranking of different unlearning methods with added standard deviations.

UL	η			
	0.1	0.4	0.8	1.0
RETRFINAL	0.340 \pm 0.017	0.586 \pm 0.015	0.621 \pm 0.014	0.634 \pm 0.025
FTFINAL	0.131 \pm 0.011	0.585 \pm 0.016	0.619 \pm 0.014	0.634 \pm 0.024
FISHER	0.751 \pm 0.024	0.679 \pm 0.005	0.734 \pm 0.006	0.791 \pm 0.020
NEGGRAD	0.124 \pm 0.010	0.564 \pm 0.018	0.603 \pm 0.014	0.656 \pm 0.035
SALUN	0.476 \pm 0.014	0.617 \pm 0.016	0.689 \pm 0.013	0.748 \pm 0.004
SSD	0.975 \pm 0.008	0.939 \pm 0.025	0.929 \pm 0.021	0.928 \pm 0.015
RETRAIN	0.999 \pm 0.000	0.997 \pm 0.001	0.993 \pm 0.001	0.993 \pm 0.001

Unlearning Quality Versus α . We provide additional experiments under different unlearning portion parameters α of the forget set to the full training set. The experiment is conducted with the CIFAR10 dataset and ResNet20 model architecture. The results are shown in Table 5. We again observe that the relative ranking of different unlearning methods stays mostly consistent across different α . Generally, the relative ranking of unlearning methods stays consistent across different α 's.

Table 5: Unlearning Quality *versus* α . The relative ranking of different unlearning methods stays mostly consistent across different α .

UL	α			
	0.1	0.25	0.4	0.67
RETRFINAL	0.513 ± 0.054	0.489 ± 0.055	0.413 ± 0.027	0.500 ± 0.024
FTFINAL	0.511 ± 0.054	0.484 ± 0.054	0.394 ± 0.041	0.479 ± 0.020
FISHER	0.904 ± 0.046	0.871 ± 0.044	0.908 ± 0.038	0.810 ± 0.050
NEGGRAD	0.637 ± 0.105	0.757 ± 0.073	0.684 ± 0.016	0.631 ± 0.088
SALUN	0.755 ± 0.017	0.762 ± 0.043	0.895 ± 0.047	0.893 ± 0.028
SSD	0.944 ± 0.038	0.960 ± 0.019	0.837 ± 0.054	0.913 ± 0.037

Unlearning Quality Versus Model Architecture. We provide additional experimental results under different model architectures. The experiment is conducted with the CIFAR10 dataset and $\alpha = 0.1$. The results are shown in Table 6. Interestingly, we observe once again that the relative ranking of different unlearning methods stays mostly consistent across different architectures.

Table 6: Unlearning Quality *versus* different model architectures. The relative ranking of different unlearning methods stays mostly consistent under different architectures.

UL	ResNet44	ResNet56	ResNet110
RETRFINAL	0.497 ± 0.040	0.473 ± 0.010	0.476 ± 0.036
FTFINAL	0.495 ± 0.041	0.471 ± 0.011	0.477 ± 0.039
FISHER	0.847 ± 0.051	0.832 ± 0.032	0.895 ± 0.020
NEGGRAD	0.562 ± 0.025	0.537 ± 0.016	0.520 ± 0.042
SALUN	0.716 ± 0.008	0.692 ± 0.013	0.672 ± 0.033
SSD	0.939 ± 0.053	0.935 ± 0.056	0.968 ± 0.017

Unlearning Quality Versus Dataset. We provide additional experiments on CIFAR100 [Krizhevsky et al., 2009] and MNIST [LeCun, 1998]. The experiment is conducted with the ResNet20 model architecture and $\alpha = 0.1$. We note that CIFAR100 has 100 classes and 50000 training images while MNIST has 10 classes and 60000 training images. CIFAR100 is considered more challenging than CIFAR10 while MNIST is considered easier than CIFAR10. The results are shown in Table 7.

In this experiment, besides the consistency we have observed throughout this section, we in addition observe that the Unlearning Quality reflects the *level of difficulties* of unlearning on different datasets. Specifically, the Unlearning Quality of most unlearning methods is higher on MNIST while lower on CIFAR100 [], in comparison to those on CIFAR10.

Table 7: Unlearning Quality *versus* different datasets. In addition to the consistency of the Unlearning Quality across unlearning methods, the Unlearning Quality scores are higher on MNIST while lower on CIFAR100, in comparison to those on CIFAR10, reflecting the level of difficulties on different datasets.

UL	CIFAR100	MNIST
RETRFINAL	0.464 ± 0.027	0.976 ± 0.020
FTFINAL	0.462 ± 0.028	0.977 ± 0.021
FISHER	0.606 ± 0.008	0.990 ± 0.002
NEGGRAD	0.669 ± 0.016	0.980 ± 0.017
SALUN	0.697 ± 0.082	0.995 ± 0.002
SSD	0.923 ± 0.058	0.998 ± 0.001

Validation of Theorem 3.5 With Linear Models We experimented with a method from Guo et al. [2020], which is an unlearning algorithm for linear models with (ϵ, δ) -certified removal guarantees.

1239 We followed the experimental setup in Guo et al. [2020], training a linear model on part of the MNIST
 1240 dataset for a binary classification task distinguishing class 3 from class 8.

1241 In their algorithm, the parameter ϵ controls a budget indicating the extent of data that can be unlearned.
 1242 During the iterative unlearning, when the accumulated gradient residual norm is beyond the unlearning
 1243 budget, the unlearning guarantee is broken and retraining will kick in. So ϵ cannot be made arbitrarily
 1244 small. Below, we report the Advantage metric for their unlearning algorithm with different ϵ (δ is
 fixed as $1e - 4$), as well as the Retrain method as reference:

ϵ	0.8	0.6	0.4	0.3	RETRAIN
Advantage	0.010	0.009	0.005	0.003	0.002

Table 8: Change of advantage in linear models with respect to decreasing ϵ .

1245

1246 We can see that the Advantage monotonically decreases as ϵ decreases, which aligns with our Theo-
 1247 rem 3.5