Abstract

We consider a practical scenario of machine unlearning to erase a target dataset, which causes unexpected behavior from the trained model. The target dataset is often assumed to be fully identifiable in a standard unlearning scenario. Such a flawless identification, however, is almost impossible if the training dataset is inaccessible at the time of unlearning. Unlike previous approaches requiring a complete set of targets, we consider few-shot unlearning scenario when only a few samples of target data are available. To this end, we formulate the few-shot unlearning problem specifying intentions behind the unlearning request (e.g., purely unlearning, mislabel correction, privacy protection), and we devise a straightforward framework that (i) retrieves a proxy of the training data via model inversion fully exploiting information available in the context of unlearning; (ii) adjusts the proxy according to the unlearning intention; and (iii) updates the model with the adjusted proxy. We demonstrate that our method using only a subset of target data can outperform the state-of-the-art unlearning methods even with a complete indication of target data.

1. Introduction

Machine unlearning is the task of excising some target dataset from a trained model. The goal is to mitigate unwanted behaviors of the model that may have been induced by the target samples, which are, e.g., mislabeled (Cao & Yang, 2015; Du et al., 2019) or privacy-sensitive (c.f., right to be forgotten (Mantelero, 2016)). While retraining a model from scratch without using the target data is a straightforward solution, it is often impossible to do so whenever the original training data is not fully accessible due to memory or privacy concerns. To this end, standard unlearning algorithms aim to approximate such oracle re-

---

1Equal contribution

1Pohang University of Science and Technology. Correspondence to: Jungseul Ok <jungseul@postech.ac.kr>.
To address this challenge, we first establish a generalized formulation of the few-shot unlearning that can take into account the intention of unlearning (Section 3). By specifying the intention, our flexible formulation can correspond to the noisy label correction or privacy protection. Also, with a complete target indication, the formulation covers the standard unlearning as a special case. Based on this formulation, we propose a straightforward framework (Section 4) consisting of (i) model inversion to retrieve a proxy for training data from all the information available in the context of unlearning including the given model and target samples (Section 4.1); (ii) filtration to divide the retrieved proxy data into a part to be preserved and the other to be unlearned or modified according to unlearner’s intention while augmenting the few target samples (Section 4.2); and then (iii) relearning with the filtered proxy data to scrub out the unwanted behavior and impose the intended one onto the augmented target dataset (Section 4.3). For the model inversion step, we design a new inversion technique specialized for unlearning. It fully utilizes the knowledge at hand in the context of unlearning: the trained model, the target samples, and the general prior on the data domain. In the relearning step, we devise a set of relearning strategies, which relabel the augmented target dataset differently from the filtration step, to impose unlearner’s intention accordingly: OurStd for standard unlearning, OurSum for privacy protection, OurNeg and OurCur for mislabeled correction.

In our experiments of standard and few-shot unlearning setups with canonical deep learning models (Section 5), we validate the superiority of our method compared to existing ones (Golatkar et al., 2020a; Chundawat et al., 2022a; Kim & Woo, 2022) with almost full access to the training and target datasets. We also demonstrate a sharp unlearning to erase only a subclass such as \( \mathcal{P} \), which is infeasible in previous methods (Chundawat et al., 2022b; Baumhauer et al., 2022) with no access to the training dataset. In particular, only our method can successfully unlearn a subset (e.g., \( \mathcal{P} \)'s) of class. Our empirical study suggests appropriate designs of unlearning methods for each of the canonical intentions: standard unlearning, mislabel correction, and privacy protection. This provides a unified view of existing works implicitly postulating different intentions, e.g., standard unlearning (Bourtoule et al., 2019; Kim & Woo, 2022) and privacy protection (Chundawat et al., 2022a). In addition, the formulation with the notion of intention allows us to newly discover a practical unlearning method OurNeg to correct the mislabeled target dataset, applicable at no extra cost of annotations.

Contributions. The main contributions of this paper can be summarized as follows:

- We establish the generalized unlearning problem that encompasses not only the few-shot unlearning scenarios but also the various unlearning intentions: standard unlearning, privacy protection, and mislabel correction.

- We devise the unlearning framework, consisting of the model inversion, filtration, and relearning steps, each of which is tailored to address few-shot unlearning with various intentions.

- In our experiment, we show the strength of our method in both few-shot and standard unlearning setups with various intentions even compared to existing methods with full access to the training and target datasets. We also corroborate that given only a trained model and few target samples, our method enables a sophisticated unlearning to eliminate a subclass, that is previously intractable.

2. Related Work

Machine unlearning. Prior works on machine unlearning can be roughly categorized into two groups according to their assumptions on the accessibility of the training dataset (denoted by \( D \)) and the target dataset (denoted by \( D_e \subset D \)). The first group assumes full access to both \( D \) and \( D_e \) (Bourtoule et al., 2019; Graves et al., 2020; Gupta et al., 2021; Tarun et al., 2021; Golatkar et al., 2020b; Chundawat et al., 2022a; Kim & Woo, 2022), and focuses on reducing the computational cost of unlearning to a level significantly below the cost of retraining a model from scratch using the remaining dataset \( D_r := D \setminus D_e \). The second one assumes no access to \( D \) except \( D_e \) (Golatkar et al., 2020a; Ye et al., 2022; Nguyen et al., 2020; Fu et al., 2021; Baumhauer et al., 2022; Chundawat et al., 2022b; Graves et al., 2020). The main focus of these works is to prevent catastrophic forgetting on \( D_r \) while erasing \( D_e \). The works in this category typically require additional information or assumption regarding the model and the training dataset, such as certain proxies of \( D \) (Golatkar et al., 2020a; Ye et al., 2022), sample-wise training history (Graves et al., 2020), or Bayesian assumptions (Nguyen et al., 2020; Fu et al., 2021). Several recent works attempt to circumvent such requirements by restricting the target dataset to be all samples of a specific class so that the remaining dataset is automatically identified by the complementary classes (Chundawat et al., 2022b; Baumhauer et al., 2022), i.e., an exquisite unlearning to erase only a subclass (e.g., \( \mathcal{P} \)) is not possible. Our few-shot unlearning method needs no such additional requirement but enables the exquisite unlearning.

Model inversion. With growing scales of machine learning datasets and increasing attention on privacy concerns, model inversion has been widely studied in various contexts of machine learning (Yoo et al., 2019; Chen et al., 2019; Yin et al., 2020; Choi et al., 2020; Luo et al., 2020; YU et al., 2021; Zhao et al., 2022; Fredrikson et al., 2015;
Few-shot Unlearning by Model Inversion

Yang et al., 2019; Jeon et al., 2021). The model inversion is particularly useful in data-free transfer learning applications, including but not limited to model compression (Yoo et al., 2019; Chen et al., 2019; Yin et al., 2020; Choi et al., 2020; Luo et al., 2020; YU et al., 2021; Zhao et al., 2022) and continual learning (Yin et al., 2020), where we want to distill the knowledge of the trained model into another but have no training data. In addition, to check privacy leakage, one can formulate the problem of inverting data from the model (Fredrikson et al., 2015; Yang et al., 2019) or even gradient (Jeon et al., 2021). In this line of works on model inversion, an extensive set of prior and side information on data domain (e.g., implicit (Rudin et al., 1992) and explicit (Ulyanov et al., 2018) image domain priors and batch norm statistics (Yin et al., 2020)) have been exploited to improve inverted data quality via reducing search space. We inherit this methodology and devise a new model inversion technique specialized under the unlearning scenario, where the target samples additionally provide hints on the training dataset. This is of independent interest to other data-free transfer learning or model inversion algorithms if few samples of training dataset are allowed.

3. Problem Formulation

For ease of exposition, we postulate a standard machine learning for image classification task with training dataset \( D \) each of which sample is a pair of image \( x \in \mathcal{X} \) and label \( y \in \mathcal{Y} \), e.g., MNIST dataset of handwritten digits. Given this dataset, a model \( f(\cdot; w) \) is parameterized by \( w \) and trained by minimizing the average cross-entropy loss \( \mathcal{L}(w; D) \) comparing \( f(x; w) \) and \( y \) over training dataset \( D \). We denote the parameter learned by this procedure by original model \( w_o \) or equivalently \( f_o := f(\cdot; w_o) \).

**Standard unlearning.** Given a subset of training samples \( D_e \subset D \) but limited access to training data \( D \), the standard unlearning aims to tune the original model \( w_o \) as if it has never seen \( D_e \) during training. Meaning that, the unlearner has access to two ingredients (Nguyen et al., 2020; Golatkar et al., 2020a): (i) \( w_o \), the original model that has been trained on the full dataset \( D \); and (ii) \( D_e \), the target dataset to be erased from the model. The goal of standard unlearning is to find a model that closely approximates the behavior of the standard oracle model that is trained only on retained dataset \( D_r := D \setminus D_e \), i.e., minimizing \( \mathcal{L}(w; D_r) \).

**Few-shot unlearning.** In this paper, we consider a more challenging but practical scenario of few-shot unlearning given target \( D_e \) and intention \( I_e \), where \( D_e \) is a subset or proxy data about the model’s behavior to be erased, and \( I_e \) inheriting the input dataset of \( D_e \), i.e., \( (x_e, y_e) \in D_e \) if \( (x_e, y_e') \in I_e \), describes an or no intention about the response to \( D_e \) after unlearning. The goal of few-shot unlearning is to erase an extension \( D'_e \) of \( D_e \) which consists of samples similar to the ones in \( D_e \), and to impose the response to \( D'_e \) as described in \( I_e \). In other words, the oracle model which our unlearning aims for is the one relearned from scratch using a modified training dataset \( D'_e \cup I_e' \) where \( D'_e := D \setminus D_e \) is the remaining data after excluding samples similar to \( D_e \) and \( I_e' \) is the extension of \( I_e \) corresponding to that from \( D_e \) to \( D'_e \), i.e., \((x'_e, y'_e) \in D'_e \) if \((x'_e, y'_e) \in I_e' \). In what follows, we describe several designs of intension \( I_e \) of interests in practice, each of which defines a different notion of oracle model trained on the corresponding \( D'_e \cup I_e' \).

**Intention for standard unlearning:** Oracle\(_{std}\). By simply setting the empty intention, i.e., \( I_e = \emptyset \), our unlearning problem becomes the standard one to erase \( D'_e \). Then, in this case, the goal is to approximate the model trained with only \( D'_e \) from scratch, called Oracle\(_{std}\). The empty intention would be selected for privacy concerns as well as correcting mislabels on the target samples.

**Intention for privacy protection:** Oracle\(_{int} \). Inspired by (Chundawat et al., 2022a), we can consider an alternative intention for privacy protection where the intended response to \( D'_e \) is an untrained model’s output. More formally, this intention is described as follows:

\[
I_e = \{(x_e, y'_e) : (x_e, y_e) \in D_e, \text{ and } y'_e = f(x_e; w_{unt})\},
\]

where \( w_{unt} \) is a randomly initialized parameter, and the corresponding oracle is denoted by Oracle\(_{int}\). Oracle\(_{int}\) is explicitly imposed the randomized responses to \( D'_e \), whereas Oracle\(_{neg}\) is not. Hence, Oracle\(_{int}\) would have less risk to expose \( D'_e \) against inversion attacks (Fredrikson et al., 2015; Yang et al., 2019) to find samples of low entropy output although it might leave a watermark giving some other clues on the target samples, discussed in Section 5.2.

**Intensions for mislabel correction:** Oracle\(_{cor} \) and Oracle\(_{neg} \). In the case that an unlearning task is requested to remove negative effects from the mislabeled dataset, we propose two intentions at different levels of supervision. When we can annotate the correct label for each sample in \( D_e \) or \( D'_e \), a strong intention is the set of \((x'_e, y'_e)\)’s such that \( y'_e \) is the correct label of \( x'_e \), and defines Oracle\(_{cor} \). To save effort for the annotation of correct labels, a weak intention can be established just from knowing that \( D_e \) is mislabeled. As negative learning (Cour et al., 2011) does, it imposes the negative supervision opposite to \( y_e \) upon \((x_e, y_e) \in D_e \), and defines Oracle\(_{neg} \).

4. Proposed Method

The proposed algorithm consists of three steps as shown in Figure 2. (i) **Model inversion** trains a generative model G
as a proxy of the original training set $D$, where we employ an extensive set of regularizers encoding all the information at hand under the unlearning scenario, including the original model, data domain prior, and target samples. (ii) Filtration first refines samples generated from $G$ to be sufficiently plausible and then partitions the refined dataset $\tilde{D}$ into $D_r$ and $D_o$, which are the sets of $D_r$-like and $D_o$-like samples, respectively. In this process, the target dataset $D_o$ is augmented. (iii) Relearning prepares $\tilde{I}_o$ by relabeling $D_o$-like samples corresponding to an unlearning intention and then adjusts the model $f$ after scratching the original model $f_o$’s response to target samples out. In the remainder of this section, we describe each step of the algorithm in more detail.

4.1. Model Inversion to Approximate Training Dataset

Recalling we have only access to the original model $f_o$ and few targets $D_o$, we need to retrieve a proxy of the entire dataset $D$ from $f_o$ but also to interpolate $D_o$ for the augmented target dataset $D'_o$. To this end, let $\mathcal{Y}^+$ be an extension of $\mathcal{Y}$ with an additional auxiliary class $y^+$ where the $D_o$-like samples are designated by class $y^+$. Then, we employ a conditional generative model $G : \mathcal{Z} \times \mathcal{Y}^+ \rightarrow \mathcal{X}$ such that, given a random noise $z \in \mathcal{Z}$ and class $y \in \mathcal{Y}^+$, $G(y, z)$ is a randomly generated image $x \in \mathcal{X}$ that is similar to (i) the target samples in $D_o$ if $y = y^+$; or (ii) the samples participated in the training of $w_o$ with label $y$ otherwise. To train such a conditional generator $G$, we utilize an extensive set of loss functions ($\ell_{tv}$, $\ell_{div}$, $\ell_{ent}$, $\ell_{aug}$, $\ell_{tar}$, and $\ell_{bn}$) to be minimized, where $\ell_{tar}$ is used only for the auxiliary class $y^+$. $\ell_{tv}$ captures our general belief on natural images. $\ell_{div}$ is a regularizer, borrowed from (Yoo et al., 2019), to prevent the mode-collapse issues and to diversify the generator’s output images. The formal definitions of $\ell_{tv}$ and $\ell_{div}$ are given in Appendix F, where we also leave a suggestion to further improve the quality of inverted samples based on the deep image prior (Ulyanov et al., 2018). In what follows, we describe the remaining losses that are induced from the training strategy of $f_o$ and the data statistics of $D_o$ or $D$.

$\ell_{ent}$ and $\ell_{aug}$ from the training strategy. Presuming a fairly successful training of $f_o$, a training sample $(x, y) \in D$ is anticipated to have a low cross entropy between the prediction $f_o(x)$ and label $y$ in most cases. We hence employ $\ell_{ent}$ comparing the predicted label $f_o(\tilde{x})$ and the conditional label $y$ of generated sample $\tilde{x} = G(z, y)$ in terms of cross entropy. It encourages the generated sample $\tilde{x} = G(z, y)$ to be predicted as the label of $D_o$ if $y = y^+$ or the class $y$ otherwise. From the fair training of $w_o$, we can also assume that the classifier $f_o$ is robust against a set $\Phi$ of data augmentation $\phi : \mathcal{X} \rightarrow \mathcal{X}$ such as flipping, rotating, and cropping, used in a training step. Indeed, such augmentations are often included in the training strategy of $f_o$. In other words, if the generated samples are similar to the training ones, then the classification result of $f_o$ on the original and augmented samples need to be similar to each other. Otherwise, they are unlikely used in training $f_o$. From this, we define a data augmentation loss $\ell_{aug}$ for each generated samples $\tilde{x} = G(z, y)$ as follows:

$$\ell_{aug}(\tilde{x}) = \sum_{\phi \in \Phi} \|f_o(\tilde{x}) - f_o(\phi(\tilde{x}))\|_2^2,$$

where the augmentation $\phi$ is possibly random.

$\ell_{tar}$ and $\ell_{bn}$ from the data statistics. To generate target-like samples, we can directly exploit the statistics about the given target $D_o$. $\ell_{tar}$ is a discrepancy measure between a set $B^+$ of generated target-like sample $G(z, y^+)$ and $D_o$. Specifically, the discrepancy is defined in the latent feature space of $f_o$ rather than the image space as we want to generate images semantically similar to $D_o$. More formally, let $f_{o,l}$ be the intermediate process up to the $l$-th layer of $f_o$. Then, for the mini-batch $B^+$ of generated target-like sample $\tilde{x}^+ = G(z, y^+)$, we define

$$\ell_{tar}(B^+) = \sum_{l} \|\mu_l(B^+) - \mu_l(D_o)\|_2^2,$$

where the summation is taken over the layers of interests and $\mu_l(B) := \frac{1}{|B|} \sum_{x \in B} f_{o,l}(x)$ is the average of the $l$-th intermediate outputs over some image set $B$. We can let $\ell_{tar}$ compare the variances of intermediate outputs but we do not as the empirical variance from the few samples is prone to distorted.

Besides the statistics of the target dataset $D_o$, that of the entire dataset $D$ is often available. Indeed, the use of the batch normalization (BN) layers is a de-facto standard architecture of deep classifier, and each BN layer represents a meta-data of $D$ in terms of the running mean $\mu_l(D)$ and variance $\sigma_l^2(D)$. Hence, when the given model $f_o$ includes such BN statistics, we can utilize them as proposed in (Yin et al., 2020). Specifically, the following BN loss $\ell_{bn}$ forces generated images to fit the statistics of $D$: for a set $B$ of generated images $\tilde{x} = G(z, y)$ with $y$ drawn from $\mathcal{Y}^+$,

$$\ell_{bn}(B) = \sum_{l: \text{BN layers}} \|\mu_l(B) - \mu_l(D)\|_2^2 + \|\sigma_l^2(B) - \sigma_l^2(D)\|_2^2,$$

where $\sigma_l^2(B)$ is the running variance of the $l$-th layer over $B$. We remark that $\ell_{bn}$ guides the generation of samples over all the conditions $\mathcal{Y}^+$ (even including $D_o$), whereas it is available only when the classifier includes BN layers. However, $\ell_{tar}$ is applicable regardless of the existence of BN layers and thus the summation of $\ell_{tar}$ in (2) can take
to further refine the generated images, we only incorporate samples whose (i) classification entropy with augmentation are lower than a pre-defined threshold $t_{\text{ent}}$; and (ii) predicted labels are consistent with and without augmentation. For those samples satisfying the two criteria, we use the predicted probability as a soft label for the generated image. Formally, the approximated dataset $\tilde{D}$ consists of $(\hat{x}, f_o(\hat{x}))$ such that: for each augmentation $\phi \in \Phi$,

$$H(f_o(\hat{x})) < t_{\text{ent}}, \quad H(f_o(\phi(\hat{x}))) < t_{\text{ent}}, \quad \text{and} \quad \argmax_y f_o(\phi(\hat{x})) = \argmax_y f_o(\hat{x}),$$

where the entropy $H(f_o(\hat{x}))$ is a measure of certainty about the prediction $f_o(\hat{x})$.

Filtration for $\tilde{D}$ and $\hat{D}_r$. Although the sample generation of $G$ can be conditioned on $y^+$ for target-like samples, we cannot partition $D$ into $\tilde{D}$ and $\hat{D}_r$ just relying on the labels used in the generation. We hence devise a distance-based filtration to identify target-like samples. Specifically, we measure the distance between the generated images and the images in $D_e$ in the latent space at the penultimate layer $f_{o,s}$ of $f_o$ instead of the input space, and then remove the images under a certain distance threshold $t_e$. Formally, the approximation $\tilde{D}_e$ of $D_e$ consists of $(x, y) \in \tilde{D}$ such that:

$$\kappa(x, \tilde{D}_e; f_{o,s}) < t_e,$$

where it directly defines $\tilde{D}_e = \tilde{D} \setminus D_e$ and $\kappa$ is a distance metric, measuring distance between a given sample $x$ and $D_e$ by maximum mean discrepancy (MMD) score. RBF kernel is employed for MMD score in this work. Noting that the nature of the exponential in RBF kernel of $\kappa$ provides a sudden change of $\kappa$ around the neighborhood of any point in $D_e$ on the feature map, we have a clear distinction of $\kappa$ values for target-like samples and others, c.f., Figure 3b. Based on this observation, we use a knee point detection mechanism (Satopaa et al., 2011) to automatically determine a proper threshold $t_e$ instead of handcrafting it.

4.3. Relearning per Unlearning Intention

We so far obtained $\tilde{D}_e$ and $\hat{D}_r$. Noting that the few-shot unlearning is defined with a specific intention as described in Section 3, we need to further prepare $\hat{I}_e$ by relabeling $\tilde{D}_e$ with the label $y'$ corresponding to the intention, i.e.,

$$\hat{I}_e := \{ (x, y') : (x, y) \in \tilde{D}_e \}.$$

Now, we describe how to implement our unlearning method for each intention associated with an oracle in Section 3:

- Ours\textsubscript{std}. For the standard unlearning, we can blindly imitate Oracle\textsubscript{std} and fine-tune from $f_o$ with only $\hat{D}_r$ discarding $\tilde{D}_e$. However, the response to target samples can be significantly biased by $f_o$, and different from
### 5. Experiment

**Scenarios.** In this section, we present the numerical analysis in the following three main unlearning scenarios, where canonical deep neural networks are trained for image classification based on MNIST (LeCun et al., 1998) and CIFAR-10 (Krizhevsky et al., 2009) datasets. Unlearning tasks are requested with 100% (full-shot) or 3% (few-shot) of $D_e$, where we omit 100% in cases of full-shot unlearning for notational simplicity. Detailed setups and more experiments carrying similar messages are postponed to the appendix.

- **-$\varphi$-noisyMNIST** scenario uses noisyMNIST where we modify MNIST such that all the samples in subclass $\varphi$ are mislabeled as class 2. A model is trained on noisyMNIST, and the target dataset $D_e$ is the mislabeled subclass $\varphi$.

- **-$\varphi$-MNIST** scenario starts with a model trained on MNIST. The unlearning task is requested to erase a subclass $\varphi$ in class 7, i.e., $D_e$ is the subclass $\varphi$.

- **$\text{Truck-CIFAR10}$** scenario begins with a model trained on CIFAR10. We want to erase the whole of truck class, while maintaining the other classes. This is one of

| Method          | $\varphi$-noisyMNIST (%) | $\varphi$-MNIST (%) | Truck-CIFAR10 (%) |
|-----------------|--------------------------|---------------------|------------------|
|                 | $D_{\varphi}$ (↑) | $D_{\varphi}$ (↑) | $D_{\varphi}$ (↑) | $D_{\varphi}$ (↑) |
| Original        | 99.4 | 0.0 | 99.7 | 98.5 | 92.7 | 96.3 |
| Oracle std      | 99.5 | 74.5 | 99.5 | 74.5 | 92.9 | 0.0 |
| Oracle unt      | 99.5 | 74.5 | 99.5 | 74.5 | 92.9 | 0.0 |
| Oracle neg      | 99.6 | 100.0 | 98.1 | 0.0 | 93.0 | 0.0 |
| Oracle cor      | 99.5 | 100.0 | 99.5 | 100.0 | 99.5 | 100.0 |
| Incompetent     | 99.5±0.5 | 91.0±1.9 | 99.5±0.1 | 17.4±7.8 | 90.3±0.7 | 0.7±1.6 |
| Neutralized     | 99.5±0.1 | 48.0±24.9 | 99.7±0.0 | 99.5±0.0 | 92.9±0.2 | 1.5±0.4 |
| Fisher          | 97.4±0.9 | 0.9±0.6 | 52.9±12.4 | 8.4±13.6 | 87.5±1.4 | 87.4±6.2 |
| Ours std        | 98.9±0.2 | 7.0±3.5 | 99.3±0.3 | 81.5±20.1 | 86.4±0.5 | 1.6±1.0 |
| Ours unt        | 98.4±0.3 | 33.3±6.0 | 98.3±0.5 | 31.1±10.3 | 87.1±0.7 | 0.4±0.2 |
| Ours neg        | 99.1±0.0 | 100.0±0.0 | 98.5±0.5 | 2.0±0.6 | 89.4±0.3 | 2.3±0.6 |
| Ours cor        | 99.3±0.0 | 100.0±0.0 | 99.3±0.0 | 100.0±0.0 | 99.3±0.0 | 100.0±0.0 |
| Ours std (3%)   | 98.5±0.9 | 5.6±2.7 | 99.6±0.1 | 90.8±2.6 | 86.4±0.7 | 4.5±1.3 |
| Ours unt (3%)   | 97.6±0.9 | 32.4±15.5 | 98.7±0.1 | 52.9±14.5 | 86.9±0.2 | 0.1±0.6 |
| Ours neg (3%)   | 99.3±0.1 | 99.7±0.4 | 98.1±0.1 | 11.7±0.5 | 89.7±0.8 | 2.3±1.4 |
| Ours cor (3%)   | 99.3±0.0 | 100.0±0.0 | 99.3±0.0 | 100.0±0.0 | 99.3±0.0 | 100.0±0.0 |

Table 1: Test accuracies on $D_e$ and $\bar{D}_e$ of unlearning methods on the representative scenarios. We report the mean with standard deviation over 5 random instances. The results in $\varphi$-noisyMNIST, $\varphi$-MNIST, and Truck-CIFAR10 are mainly discussed in Section 5.1, 5.2, and 5.3, respectively.
the simplest unlearning scenarios. However, it is much more challenging than a similar one based on MNIST (grayscale) due to its higher complexity of data domain (colored) in particular when we have no free access to \( D \).

**Baselines.** We compare the models from a set of unlearning methods, described in what follows:

- **Original** denotes the model \( f_o \) trained with \( D \) in advance of unlearning, i.e., no unlearning.
- **Oracle** is an ideal reference model trained with complete \( D \) and modified \( D_e \) for each task from the scratch, with some specific intention as described in Section 3.
- **Ours** is from our few-shot unlearning method with the generator trained from scratch in Section 4.1. An unlearning method can be numerically assessed by measuring the discrepancy between the behaviors of the oracle and its output model, whereas a better unlearning method has a smaller discrepancy to the oracle of the context of interests, e.g., for mislabel correction, Oracle_{cor} would be compared to Ours_{cor} or Ours_{neg}.
- **Incompetent, Neutralized, and Fisher**, are the unlearning methods proposed by Chundawat et al. (2022a), Kim & Woo (2022), and Golatkar et al. (2020a), respectively, and dedicated to Oracle_{ent}, Oracle_{std}, and Oracle_{std}, respectively. We note that they require practically free access to \( D \) although they are applicable to canonical deep neural networks.

### 5.1. Mislabel Correction

**Ours_{cor} and Ours_{neg} for correction.** The first column of Table 1 compares unlearning methods under \( \#\text{-noisyMNIST} \) scenario of which the ideal objective is Oracle_{cor} that corrects Original’s misbehavior induced from the mislabeled subclass \( \#\). The objective of Oracle_{cor} is closely achieved by both Ours_{cor} and Ours_{neg}, which drastically improved the low accuracy of Original on \( D_e \) while maintaining the high accuracy on \( D_r \). Remarkably, Ours_{cor} and Ours_{neg} given only 3\% of \( D_e \) (24 images of \( \#\)) also attain the ideal model correction, closely. Recalling Ours_{cor} requires additional efforts to collect the correct annotations of \( D_e \) but Ours_{neg} does not, this suggests that Ours_{neg} is a particularly practical unlearning method for mislabel correction.

**Necessity of more than erasing for correction.** In the experiment of \( \#\text{-noisyMNIST} \), we also observe a substantial gap between Oracle_{cor} and Oracle_{std} (or others dedicated to it: Incompetent, Fisher, and Ours_{std}) even if all the mislabeled target samples are indicated. This gap implies that more than just erasing is necessary to remedy the misbehavior induced by \( D_e \). Otherwise, the response on \( D_r \) becomes unpredictable after unlearning, c.f., the oscillating response to \( D_e \) of Oracle_{std} in Appendix B.

### 5.2. Privacy Protection

The second column of Table 1 provides a comparison of unlearning methods in \( \#\text{-MNIST} \) of which the unlearning request can be asked for privacy protection of subclass \( \#\).

An ideal unlearning result is to retain Original’s accuracy on \( D_r \) (including non-crossed 7’s) and to have an equivocal prediction on \( D_e \) (sub-class \( \#\)) so that the samples in \( D_e \) are not easily identifiable by the model response. In this sense, only Neutralized fails to hide \( D_e \) since its accuracy on \( D_e \) is as high as Original. However, this analysis is insufficient to identify suitable unlearning methods for preserving privacy.

**Membership inference attack.** To further investigate unlearning to protect data privacy, we borrow a measure of privacy leakage which is attack success rate \( (\text{ASR}) \) of a membership inference method (Shokri et al., 2017), also used in (Golatkar et al., 2020b). To be specific, a binary support vector \( (\text{SVM}) \) taking the model \( f \)’s output to a sample is trained to infer whether or not the sample is used in the training of \( f \). Noting that a privacy leakage can be conducted by model inversion to retrieve inputs having a distinguishable output of \( f \) (one-hot or low-entropy vector), we can anticipate that the SVM identifies the samples that have \( f \)’s distinguishable outputs implying high risks of model inversion attack success. In this sense, low ASR on \( D_e \) is a necessary condition for preserving privacy on \( D_e \).

**Necessity of more than erasing (Ours_{unt}) for privacy.** Table 2 first compares the ASR’s of unlearning methods in \( \#\text{-MNIST} \) scenarios. This comparison suggests Oracle_{unt} as a reference oracle for privacy protection. Hence, Incompetent and Ours_{unt} show low ASR’s on \( D_e \) as they aim at imitating Oracle_{unt}. Again, Ours_{unt} fairly protects \( D_e \)’s privacy even when 3\% of \( D_e \) is given. As expected from the analysis based on accuracy, Neutralized has a high ASR. It is also interesting that Oracle_{std} completely fails while Ours_{std} reduces ASR on \( D_e \). However, the huge gap between ASRs of Ours_{unt} and Ours_{std} verifies the necessity of more than just erasing for privacy protection.

**Watermark after unlearning.** In Table 2, we also report L2 norm average of the model \( f \)’s penultimate outputs to \( D_e \) and \( D_r \). Interestingly, Oracle_{unt}, Ours_{unt} and Incompetent, which successfully make the (ultimate) outputs to \( D_e \) indistinguishable, have noticeably small L2 norm on \( D_e \), i.e., the penultimate outputs to \( D_e \) around the origin. From this fact, it is possible to design a new model inversion attack to retrieve the unlearned samples by finding input data such that its feature on the penultimate layer is close to the origin, i.e., the untrained intention carves a watermark in the penultimate layer.
Few-shot Unlearning by Model Inversion

Figure 3: (a) Generated images in $\tilde{D}$, where the target-like samples $D_e$ identified by the filtration step are marked with the red boxes; and (b) sorted MMD of a generated sample in $\tilde{D}$ to $D_e$, where the horizontal line indicates the threshold $t_l$ for the filtration found by knee point detector. (left: $\mathcal{F}$-MNIST; and right: Truck-CIFAR10)

| Method       | ASR (%) | L2 norm |
|--------------|---------|---------|
|              | $D_e$   | $D_e(\uparrow)$ | $D_r$ | $D_e$ |
| Original     | 100.0   | 100.0   | 20.6±3.4 | 21.9±3.2 |
| Oracle$_{std}$ | 100.0   | 100.0   | 13.5±1.0 | 12.5±0.8 |
| Oracle$_{unt}$ | 99.5    | 0.0     | 12.3±0.9 | 7.7±0.2 |
| Oracle$_{neg}$ | 100.0   | 100.0   | 13.0±0.9 | 12.0±0.4 |
| Incompetent  | 99.4    | 23.8    | 24.0±4.6  | 6.1±0.9 |
| Neutralized  | 100.0   | 100.0   | 23.2±8.9  | 20.0±3.0 |
| Fisher       | 100.0   | 100.0   | 19.5±3.2  | 17.7±2.4 |
| Ours$_{std}$  | 99.7    | 82.5    | 19.2±4.9  | 11.2±1.2 |
| Ours$_{unt}$  | 99.7    | 30.0    | 17.4±6.3  | 4.2±1.0 |
| Ours$_{neg}$  | 99.8    | 100.0   | 21.7±4.3  | 19.3±2.8 |
| Ours$_{std}$ (3%)  | 100.0   | 85.0    | 20.0±4.4  | 12.4±1.8 |
| Ours$_{unt}$ (3%)  | 100.0   | 32.5    | 18.0±6.6  | 4.1±1.2 |
| Ours$_{neg}$ (3%)  | 100.0   | 100.0   | 21.8±4.2  | 18.8±2.5 |

Table 2: Attack success rate (ASR) and averaged L2 norm of the penultimate features in $\mathcal{F}$-MNIST. Attack Success Rate reports the result of the membership inference attack method. L2 norm shows watermark effect mentioned in Section 5.2, that depending on unlearning method, $D_e$‘s norm in latent space can be significantly small which is an evidence that such samples are unlearned.

5.3. Design Justification

In what follows, we provide empirical justifications for our design choices.

Challenging model inversion. The model inversion of Truck-CIFAR10 is more challenging than that of $\mathcal{F}$-MNIST due to the increased complexity of input domain (grayscale → RGB). However, as shown in Figure 3, our inversion method produces reasonable proxies for both $\mathcal{F}$-MNIST and Truck-CIFAR10. Also, the last column of Table 1 shows that even in Truck-CIFAR10 and given a few targets, our unlearning method with each intention successfully unlearns $D_e$ as much as the corresponding oracle does, although the accuracy on $D_r$ is slightly degenerated after unlearning. The quality and diversity of images generated by the inversion could be one reason for the performance degeneration.

Key loss in model inversion. Recalling that in $\mathcal{F}$-noisyMNIST we can assess the quality of unlearning clearly in terms of the accuracy on $D_r$ and $D_e$ (higher is better), Table 3 presents an ablation study of the key losses ($\ell_{\text{tar}}$, $\ell_{\text{aug}}$, and $\ell_{\text{bn}}$) in our model inversion method by evaluating the downstream unlearning performance. Apparently, $\ell_{\text{tar}}$, which is a novel loss specialized for the unlearning context, covers the most gain from $\ell_{\text{aug}}$ and $\ell_{\text{bn}}$. In a typical model inversion, $\ell_{\text{bn}}$ is known to be the most prominent (Yin et al., 2020). Hence, the ablation study suggests that even when $\ell_{\text{bn}}$ is inapplicable due to no BN layers, we can get similar results by actively utilizing $\ell_{\text{tar}}$. More ablation studies about normalization methods can be found in Appendix D.2.

Filtration with knee point detection. When dividing $\tilde{D}$ into $D_e$ and $D_r$, the threshold $t_l$ in (3) plays a critical role. For an automatic yet robust choice of $t_l$, we use the knee point detection mechanism (Satopaa et al., 2011). As shown in Figure 3b, there is a knee point in the cumulative distribution of the MMD, and also the knee point separates $\tilde{D}_e$ and $\tilde{D}_r$ properly for both $\mathcal{F}$-MNIST and Truck-CIFAR10. Nevertheless, as ours can observe generated samples and whether it is filtered or not, the threshold $t_l$ can be chosen manually if the unlearner wants to.

6. Conclusion

We formulated the comprehensive unlearning problem that covers a wide range of scenarios, including those with few
| Method   | Accuracy (%) | $D_r$ (↑) | $D_e$ (↑) |
|----------|--------------|-----------|-----------|
| Original | 99.4         | 0.0       |           |
| Oracle$_{cor}$ | 99.5     | 100.0     |           |

| $\ell_{tar}$ | $\ell_{aug}$ | $\ell_{bn}$ |
|--------------|--------------|--------------|
| ✓            | 98.5±0.1     | 0.3±0.7      |
| ✓ ✓          | 93.9±1.0     | 98.1±0.7     |
| ✓ ✓ ✓        | 95.7±0.4     | 99.9±0.3     |
| ✓ ✓ ✓ ✓      | 99.3±0.0     | 100.0±0.0    |

Table 3: Ablation study for the prior information ($D_e$, augmentation, and batch normalization statistics) on $\hat{\Phi}$-noisyMNIST.

target examples, as well as various purposes: standard unlearning, privacy protection, and mislabel correction. For this, we proposed the framework, composed of the model inversion, filtration, and relearning steps. Our experiment corroborated the novelty and superiority of each step in the representative yet diverse unlearning scenarios, including the subclass and few-shot unlearning previously infeasible. It also suggested the necessity of more than erasing targets to attain a specific unlearning purpose such as privacy protection or mislabel correction.
References

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

Bourtoule, L., Chandrasekaran, V., Choquette-Choo, C. A., Jia, H., Travers, A., Zhang, B., Lie, D., and Papernot, N. Machine unlearning. *arXiv preprint arXiv:1912.03817*, 2019.

Cao, Y. and Yang, J. Towards making systems forget with machine unlearning. In *2015 IEEE Symposium on Security and Privacy*, pp. 463–480. IEEE, 2015.

Chen, H., Wang, Y., Xu, C., Yang, Z., Liu, C., Shi, B., Xu, C., Xu, C., and Tian, Q. Data-free learning of student networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3514–3522, 2019.

Choi, Y., Choi, J., El-Khamy, M., and Lee, J. Data-free network quantization with adversarial knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 710–711, 2020.

Chundawat, V. S., Tarun, A. K., Mandal, M., and Kankanhalli, M. Can bad teaching induce forgetting? Unlearning in deep networks using an incompetent teacher. *arXiv preprint arXiv:2205.08096*, 2022a.

Chundawat, V. S., Tarun, A. K., Mandal, M., and Kankanhalli, M. Zero-shot machine unlearning. *arXiv preprint arXiv:2201.05629*, 2022b.

Cour, T., Sapp, B., and Taskar, B. Learning from partial labels. *Journal of Machine Learning Research*, 2011.

Du, M., Chen, Z., Liu, C., Oak, R., and Song, D. Lifelong anomaly detection through unlearning. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1283–1297, 2019.

Fredrikson, M., Jha, S., and Ristenpart, T. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pp. 1322–1333, 2015.

Fu, S., He, F., Xu, Y., and Tao, D. Bayesian inference forgetting. *arXiv preprint arXiv:2101.06417*, 2021.

Golatkar, A., Achille, A., and Soatto, S. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9304–9312, 2020a.

Golatkar, A., Achille, A., and Soatto, S. Forgetting outside the box: Scrubbing deep networks of information accessible from input-output observations. In *European Conference on Computer Vision*, pp. 383–398. Springer, 2020b.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems*, 2014.

Graves, L., Nagisetty, V., and Ganesh, V. Amnesiac machine learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

Gupta, V., Jung, C., Neel, S., Roth, A., Sharifi-Malvajerdi, S., and Waites, C. Adaptive machine unlearning. *arXiv preprint arXiv:2106.04378*, 2021.

He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV*, pp. 630–645. Springer, 2016.

Jeon, J., Kim, J., Lee, K., Oh, S., and Ok, J. Gradient inversion with generative image prior. In *Advances in Neural Information Processing Systems*, 2021.

Kim, J. and Woo, S. S. Efficient two-stage model retraining for machine unlearning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 4361–4369, June 2022.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Luo, L., Sandler, M., Lin, Z., Zhmoginov, A., and Howard, A. Large-scale generative data-free distillation. *arXiv preprint arXiv:2012.05578*, 2020.

Mantelero, A. Right to be forgotten and public registers – A request to the European court of justice for a preliminary ruling. *Eur. Data Prot. L. Rev.*, 2:231, 2016.

Nguyen, Q. P., Low, B. K. H., and Jaillet, P. Variational Bayesian unlearning. *Advances in Neural Information Processing Systems*, 33, 2020.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.
Rudin, L. I., Osher, S., and Fatemi, E. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.

Satopaa, V., Albrecht, J., Irwin, D., and Raghavan, B. Finding a “kneedle” in a haystack: Detecting knee points in system behavior. In *2011 31st International Conference on Distributed Computing Systems Workshops*, pp. 166–171. IEEE, 2011.

Shokri, R., Stronati, M., Song, C., and Shmatikov, V. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pp. 3–18. IEEE, 2017.

Tarun, A. K., Chundawat, V. S., Mandal, M., and Kankanhalli, M. Fast yet effective machine unlearning. *arXiv preprint arXiv:2111.08947*, 2021.

Ulyanov, D., Vedaldi, A., and Lempitsky, V. Deep image prior. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2018.

Yang, Z., Chang, E.-C., and Liang, Z. Adversarial neural network inversion via auxiliary knowledge alignment. *arXiv preprint arXiv:1902.08552*, 2019.

Ye, J., Fu, Y., Song, J., Yang, X., Liu, S., Jin, X., Song, M., and Wang, X. Learning with recoverable forgetting. In *European Conference on Computer Vision*, pp. 87–103. Springer, 2022.

Yin, H., Molchanov, P., Alvarez, J. M., Li, Z., Mallya, A., Hoiem, D., Jha, N. K., and Kautz, J. Dreaming to distill: Data-free knowledge transfer via deepinversion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8715–8724, 2020.

Yoo, J., Cho, M., Kim, T., and Kang, U. Knowledge extraction with no observable data. *Advances in Neural Information Processing Systems*, 32, 2019.

YU, X., Yan, L., and Ou, L. Conditional generative data-free knowledge distillation based on attention transfer. *arXiv preprint arXiv:2112.15358*, 2021.

Zhao, H., Sun, X., Dong, J., Manic, M., Zhou, H., and Yu, H. Dual discriminator adversarial distillation for data-free model compression. *International Journal of Machine Learning and Cybernetics*, 13(5):1213–1230, 2022.
A. Implementation Details

Model architecture. For all of our experiments, we use a classifier model architecture, PreAct-ResNet18 (He et al., 2016) as the prior works (Golatkar et al., 2020a; Chundawat et al., 2022b) use similar ResNet architectures. As our method works even if it does not have BatchNorm layer as shown in Section D.2, no specific architecture is required. Our generator architecture is inspired by DCGAN (Radford et al., 2015) which has quite simple architecture. A more complex model could generate better images, but such a simple model still shows promising unlearning performance in experiments.

Detail of $\ell_{\text{tar}}$. In the implementation of $\ell_{\text{tar}}$, only batch normalization layers are selected to calculate the mean values of each batch, as done in $\ell_{\text{bn}}$. However, unlike $\ell_{\text{bn}}$, it is not restricted to use batch normalization layer only and can be extended to all layers.

B. Ill-Determined Standard Unlearning

Oraclestd, trained only with $D_r$ (and without any intention), does not clarify the expected behavior on $D_e$, which can be under-defined and unpredictable. Thus, we conduct an experiment to show the predictions on $D_e$ in Oraclestd on $\mathbb{P}$-MNIST scenario. In Figure 4, we report the predicted labels of $D_e$ over training epochs. With a big learning rate or simple model architecture in Figure 4b and Figure 4c, instability of prediction on $\mathbb{P}$ is shown. Additionally, unlearning 9 shows more vibration, since 9 is similar to both 4 and 7. Thus, lack of expected behavior on $D_e$ leads to an under-defined oracle and uncertain objective for unlearning, where the intention of the user becomes important.

![Figure 4: Predictions for $D_e$ of Oraclestd. The predictions often oscillate over training epochs. This shows that the standard unlearning is indeed ill-determined on the model response to the unseen data in $D_e$.](image-url)
Few-shot Unlearning by Model Inversion

| Method     | 9-MNIST (%) | 8-MNIST (%) | 7-MNIST (%) |
|------------|-------------|-------------|-------------|
|            | $D_r(\uparrow)$ | $D_r(\downarrow)$ | $D_r(\uparrow)$ | $D_r(\downarrow)$ | $D_r(\uparrow)$ | $D_r(\downarrow)$ |
| Original   | 99.7        | 99.3        | 99.7        | 99.3        | 99.7        | 99.3        |
| Oraclestd  | 99.5        | 0.0         | 99.5        | 0.0         | 99.5        | 0.0         |
| Oracleant  | 99.6        | 0.0         | 99.3        | 0.0         | 99.4        | 0.0         |
| Oracleneg  | 99.5        | 0.0         | 99.4        | 0.0         | 99.4        | 0.0         |
| Incompetent| 99.4±0.1    | 0.0±0.0     | 99.5±0.2    | 0.0±0.0     | 99.4±0.2    | 0.0±0.0     |
| Neutralized| 99.7±0.0    | 0.0±0.0     | 99.5±0.1    | 0.0±0.0     | 98.9±0.1    | 0.0±0.0     |
| Fisher     | 97.5±2.4    | 96.4±1.0    | 97.9±1.3    | 87.3±1.4    | 98.1±1.0    | 97.0±1.8    |
| Oursstd    | 98.5±0.1    | 0.0±0.0     | 98.4±0.2    | 0.0±0.0     | 98.3±0.3    | 0.0±0.0     |
| Oursant    | 98.5±0.4    | 0.0±0.0     | 98.4±0.2    | 0.7±1.6     | 97.4±0.5    | 0.2±0.2     |
| Oursneg    | 98.6±0.2    | 0.0±0.0     | 98.7±0.3    | 0.0±0.0     | 98.8±0.4    | 0.0±0.0     |
| Oursstd (3%)| 98.5±0.2    | 0.0±0.0     | 97.9±0.6    | 0.0±0.0     | 98.3±0.4    | 1.2±2.8     |
| Oursant (3%)| 98.1±0.5    | 0.0±0.0     | 98.1±0.2    | 0.2±0.2     | 97.3±1.1    | 1.0±1.1     |
| Oursneg (3%)| 98.8±0.1    | 0.0±0.0     | 98.3±0.3    | 0.0±0.0     | 99.1±0.2    | 0.3±0.3     |

Table 4: Unlearned models for unlearning each class of 9, 8, and 7 in MNIST, denoted by 9-MNIST, 8-MNIST and 7-MNIST, respectively. We report the mean with standard deviation over 5 random instances.

C. Unlearning a class

We evaluate our algorithm on the class unlearning task, which unlearns a whole class (each of 9, 8, 7) in the MNIST as done in Section 5.2. Our overall results of baseline methods are in Table 4, all Oracles show similar performance on both $D_r$ and $D_c$ accuracy. Other baseline methods showed similar performance to Oracle, while Fisher showed high $D_r$ Accuracy. We note that reported previous methods have full training dataset $D$, while our method only uses target data $D_c$. Our methods reported slightly less than them but completely erased $D_c$ while maintaining high $D_r$ accuracy. In addition, our methods achieved performance similar to the oracles on a few-shot setting (3%) which shows that our method works robustly even in small samples. Ablation study on various ratios of $D_c$ is shown in Appendix D.1.

D. Ablation Studies

D.1. Ablation study on the amount of given $D_c$

Here, we study the ratio of given $D_c$; 100%, 30%, 10%, 3% and single image. Oracle reported results based on the ratio of $D_r$, which means the remaining ratio is included in $D_r$. The experiments are conducted in the setting from Section 5.2, which unlearns 7 in MNIST. Table 5 shows that all three of our methods relatively robustly keep the accuracy of $D_r$. The cases of Oursstd and Oursant do not show a tendency according to the amount of De because oracle is not stable, as shown in Section B. Oursneg shows high accuracy on $D_r$ when few samples are used, as the filter failed to filter many $\hat{z}$, which leads to learn $\hat{z}$ as 7. However, Oraclestd, Oracleant, and Oracleneg show similar accuracy with Original when only portion of $D_c$ is given, which assumes remaining target data is included in $D_r$.

D.2. Ablation study on normalization methods

Table 3 shows that $\ell_{hn}$ is the strong performance-enhancing loss in our method. However, recently, models that use various normalization methods such as LayerNorm and InstanceNorm are also been widely used. Thus, we conduct the experiment on the other normalized method without $\ell_{hn}$, which follows the same setting with Section C for the Oracle neg and Oursneg. To compare the normalization method only, we just change the normalization layer of the model architecture from BatchNorm to LayerNorm or InstanceNorm. As shown in Table 6, our methods had a slight drop in performance on $D_r$ but can erase almost all $D_c$ with even other normalization methods.
Table 5: Unlearning performance at different $D_e$ amounts. We report the result of $\mathcal{F}$-MNIST in Section 5.2 to erase the subclass $\mathcal{F}$ in MNIST.

### D.3. Ablation study on scrubbing

For standard unlearning by retraining, Kim & Woo (2022) show that scrubbing before retraining decreases unlearning epochs. Thus, we study scrubbing for Ours$_{std}$. The experiments setting follows Section C, 9-MNIST. Table 7 shows that Ours$_{std}$—scrub still remember $D_e$ even after unlearning, which emphasizes that scrubbing is essential for Ours$_{std}$ to erase $D_e$.

### E. Privacy on Class unlearning

In Table 8, we report the result of ASR and L2 norm on 9-MNIST scenario. Similar to Table 2, Oracle$_{unt}$, Incompetent, Ours$_{std}$, and Ours$_{unt}$ show relatively small ASR than others. Moreover, overall ASR in Table 8 is smaller than that in Table 2, as it requires erasing a large amount of information. For the same reason, the difference in L2 norms between $D_r$ and $D_e$ also widened.
Table 6: Unlearning performance for different normalization methods on the task which unlearning 9 in MNIST. All of
the experiments are conducted in a negative setting. *BN, *LN, and *IN are the methods when the baseline classifier uses
BatchNorm, LayerNorm, and Instance Norm.

| Method | Accuracy (%) |
|--------|--------------|
|        | $D_r$ | $D_e$ |
| Original_{BN} | 99.71 | 99.31 |
| Oracle_{BN} | 99.52 | 0.00 |
| Ours_{BN} | 98.61±0.23 | 0.00±0.00 |
| Ours_{BN}(3%) | 98.80±0.14 | 0.00±0.00 |
| Original_{LN} | 99.31 | 98.81 |
| Oracle_{LN} | 99.52 | 0.00 |
| Ours_{LN} | 97.95±0.24 | 0.00±0.00 |
| Ours_{LN}(3%) | 97.76±0.51 | 0.00±0.00 |
| Original_{IN} | 99.11 | 99.69 |
| Oracle_{IN} | 99.59 | 0.00 |
| Ours_{IN} | 97.51±0.78 | 0.06±0.05 |
| Ours_{IN}(3%) | 96.35±0.47 | 1.19±0.56 |

Table 7: Unlearning performance without scrubbing for Ours_{std} on 9-MNIST scenario. Ours_{std−scrub} denotes Ours_{std} without
scrubbing.

| Method | Accuracy (%) |
|--------|--------------|
|        | $D_r$ | $D_e$ |
| Original | 99.71 | 99.31 |
| Oracle_{std} | 99.52 | 0.00 |
| Ours_{std} | 98.53±0.11 | 0.00±0.00 |
| Ours_{std−scrub} | 99.18±0.21 | 96.14±1.37 |
| Ours_{std}(3%) | 98.53±0.15 | 0.00±0.00 |
| Ours_{std−scrub}(3%) | 99.35±0.06 | 96.71±0.41 |

F. Formal Definition of $\ell_{tv}$ and $\ell_{div}$

Here, we give a formal definition of losses used for model inversion, $\ell_{tv}$ and $\ell_{div}$ from 4.1.

**Diversity loss.** GANs often suffer from mode-collapsing (Goodfellow et al., 2014). In our case, having a few collapsed
samples per label can potentially make the unlearning unstable since the coverage of the generated samples is limited.
To overcome the issue of mode-collapse, we propose a modification of diversity loss proposed in (Yoo et al., 2019).
Let $f_e : \mathcal{X} \rightarrow \mathbb{R}^D$ be a feature extractor, which is the target classifier except for the last fully connected layer. Our diversity
loss is defined as

$$\ell_{div}(B) = \exp \left( - \sum_{(z_i, y) \in B} \| z_i - z_j \|_2 \cdot d_{f_e} \right), \quad (5)$$

where $z_i, z_j$ are two randomly sampled noises, and $d_{f_e}$ denotes the feature-wise distance between two generated samples,
where the distance is measured with $\ell_1$ norm. The feature is an output of convolution layers of the target classifier. In
(Yoo et al., 2019), the pixel-wise distance between $G(z, y)$ is used instead of the feature-wise distance.

**Total variation loss.** One of the common prior on natural images is that they have local smoothness, i.e., a small total
| Method         | ASR (%) | L2 norm |
|----------------|---------|---------|
|                | $D_e$   | $D_{de}$| $D_e$   | $D_{de}$ |
| Original       | 100.0   | 100.0   | 20.84±3.34 | 18.40±2.88 |
| Oracle_std     | 100.0   | 100.0   | 22.10±4.33 | 15.45±1.76 |
| Oracle_unt     | 98.4    | 3.2     | 25.04±6.94 | 5.05±0.59  |
| Oracle_neg     | 100.0   | 100.0   | 24.19±4.20 | 15.92±2.00 |
| Incompetent    | 98.3    | 12.8    | 17.57±5.25 | 2.27±1.17  |
| Neutralized    | 99.9    | 55.3    | 22.59±4.45 | 13.40±2.12 |
| Fisher         | 100.0   | 100.0   | 19.54±3.20 | 17.72±2.44 |
| Ours_std       | 99.7    | 19.8    | 23.44±4.89 | 7.34±2.04  |
| Ours_unt       | 99.9    | 13.2    | 23.79±5.87 | 4.80±2.57  |
| Ours_neg       | 99.8    | 60.2    | 20.44±3.61 | 11.77±2.00 |
| Ours_std (3%)  | 99.7    | 19.9    | 21.21±4.29 | 7.71±2.53  |
| Ours_unt (3%)  | 99.9    | 16.8    | 21.88±5.85 | 4.38±2.33  |
| Ours_neg (3%)  | 99.9    | 55.0    | 19.57±4.94 | 10.74±2.40 |

Table 8: Attack Success Rate and L2 norm results to check privacy on De for unlearning class 9 in MNIST. Attack Success Rate reports the result of the membership inference attack method. L2 norm is calculated on the latent space.

To enforce the local smoothness of a generated image, we have used the total variation loss defined as:

$$\ell_{tv}(B) = \sum_{(z,y) \in B} \sum_{(i,j),(i',j') \in \delta(i,j)} \| \tilde{x}(i,j) - \tilde{x}(i',j') \|_2^2,$$

where $\delta(i,j)$ indicates a set of pixels adjacent to pixel $(i,j)$. The subscript indexes the generated pixel at a corresponding location.