Boundary Unlearning: Rapid Forgetting of Deep Networks via Shifting the Decision Boundary

Anonymous CVPR submission

Paper ID 7156

Abstract

The practical needs of the "right to be forgotten" and poisoned data removal call for efficient machine unlearning techniques, which enable machine learning models to unlearn, or to forget a fraction of training data and its lineage. Recent studies on machine unlearning for deep neural networks (DNNs) attempt to destroy the influence of the forgetting data by scrubbing the model parameters. However, it is prohibitively expensive due to the large dimension of the parameter space. In this paper, we refocus our attention from the parameter space to the decision space of the DNN model, and propose Boundary Unlearning, a rapid yet effective way to unlearn an entire class from a trained DNN model. The key idea is to shift the decision boundary of the original DNN model to imitate the decision behavior of the model retrained from scratch. We develop two novel boundary shift methods, namely Boundary Shrink and Boundary Expanding, both of which can rapidly achieve the utility and privacy guarantees. We extensively evaluate Boundary Unlearning on CIFAR-10 and Vggface2 datasets, and the results show that Boundary Unlearning can effectively forget the forgetting class on image classification and face recognition tasks, with an expected speed-up of $17\times$ *and* $19\times$*, respectively, compared with retraining from the scratch.*

1. Introduction

Suppose a company trains a face recognition model with your photos and deploys it as an opened API. Your photos could be stolen or inferenced by attackers via model inversion attack [6,18]. With the increasing awareness of protecting user's privacy, a lot of privacy regulations take effect to provide you the control over your personal data. For examples, the General Data Protect Regulation (GDPR) established by the European Union gives individuals "*the right to be forgotten*" and mandates that companies have to erase personal data once it is requested [35].

Beyond the "right to be forgotten", data forgetting from machine learning (ML) models is also beneficial when certain training data becomes no longer valid, e.g., some training data is manipulated by data poisoning attacks [10, 26], or outdated over time, or even identified to be mistakes after training. These practical needs call for efficient *machine unlearning* techniques, which enable ML models to unlearn, or to forget a fraction of training data and its lineage.

In this paper, we focus on unlearning an entire class from deep neural networks (DNNs), which is useful in realistic scenarios like face recognition: unlearning one's data needs to forget the entire class of one's face images. As the DNN model retrained from scratch is the optimal unlearned model, early studies try to accelerate the retraining process of deep networks [1, 11, 38], but have to intervene the original training process, which degenerates the model utility and increases the training time. A branch of recent researches [8, 9, 23, 27] attempt to destroy the influence of the forgetting data by scrubbing the model parameters. For example, the Fisher Information Matrix (FIM) is used to locate the influence of forgetting data at the parameter space [8, 9]. However, it is prohibitively expensive due to the large dimension of the parameter space.

In order to find an efficient unlearning approach to forget an entire class, we visualize the decision space of the retrained DNN model and discover two key observations (c.f. Figure 1). First, the forgetting samples spread around the decision space of the retrained DNN model, indicating that the decision boundary of the forgetting samples has been broken. Second, most of the forgetting samples move to the border of other clusters; this helps us recall the closest-to-boundary criterion [24] that samples at the border of cluster in the decision space will probably be predicted with huge uncertainty.

These two observations naturally match the two critical goals of machine unlearning: utility and privacy guarantees. Utility guarantee ensures that the unlearned model should generalize badly on the forgetting data while the prediction performance on the remaining data is maintained. Privacy guarantee means that the unlearned model should not leak any information of the forgetting data. Based on our key ob-

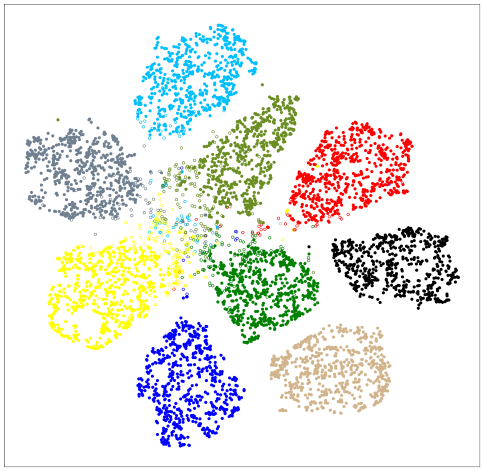


Figure 1. Key observations from the decision space of the retrained DNN model. The solid dots in different colors represent the remaining samples belonging to different classes and the hollow circles in different colors stand for the forgetting samples predicted as corresponding classes. It can be observed that (1) the forgetting samples spread around the feature space of the retrained DNN model, and (2) most of the forgetting samples move to the borders of other clusters.

servations, the utility guarantee can be achieved by only destroying the boundary of the forgetting class but maintaining the boundary of the remain classes, while the privacy guarantee can be accomplished by pushing the forgetting data to the border of other clusters.

In light of the above ideas, we refocus our attention from the parameter space to the decision space of the DNN model, and propose *Boundary Unlearning*, a rapid yet effective way to unlearn the forgetting class from a trained DNN model. Boundary Unlearning tries to shift the decision boundary of the original DNN model to imitate the decision behavior of the retrained model. To achieve the critical goals, we further introduce two novel boundary shift methods: *Boundary Shrink* and *Boundary Expanding*. The former breaks the decision boundary of the forgetting class by splitting the forgetting feature space into other classes, while the latter disperses the activation about the forgetting class by remapping and pruning an extra shadow class assigned to the forgetting data.

We summarize our major contributions as follows:

- We propose Boundary Unlearning, the first work to unlearn an entire class from a trained DNN model by shifting the decision boundary. Compared with existing studies, Boundary Unlearning neither costs too much computational resource nor intervenes the original training pipeline.
- We propose two novel methods, namely, Boundary Shrink and Boundary Expanding, to shift the decision boundary of the forgetting class. Both methods can rapidly achieve the utility and privacy guarantees with only a few epochs of boundary adjusting.
- We conduct extensive experiments to evaluate Boundary Unlearning on image classification and face recognition tasks. The results show that Boundary Unlearning can rapidly and effectively forget the forgetting class, and outperforms four state-of-the-art techniques. The code has been released for reproducibility[1].

2. Related Work

Machine unlearning is first introduced on convex models to find some comprehensible indications, but is found more useful and challenging for DNNs. Existing unlearning methods for DNNs can be broadly categorized into two groups: retrain acceleration and updating parameters.

Unlearning for ML Models. Machine unlearning is first proposed in statistical query learning, which makes the trained model forget specific data by transforming learning algorithm into a summation form [4]. Obviously, not every ML algorithm can be converted to summation form. So a few works [2, 7, 12] explore the unlearning task on convex models and try to find some constructive indications with solid theoretical bases. However, they cannot generalize to DNNs due to the non-convex nature of the loss functions.

Retrain Acceleration for DNN Models. The DNN model retrained from scratch with the remaining data is a naive yet optimal unlearning method, but it is expensive in terms of the cost of training time and resources. A few approaches are thus proposed to accelerate the retrain process. An approach called SISA adopts the idea of "divide and conquer" and proposes to split and mark the training data to train some sub-models and then aggregates them into the final well-trained model [1]. Graves *et al*. [11] achieve retrain acceleration by saving the gradient information of each training batch during the original training phase and unlearn the forgetting data by subtracting the gradient update of the specific batch. Similarly, Wu *et al*. [38] propose Deltagrad, which saves and subtracts the gradient of each forgetting sample by leveraging Quasi-Newton method. More recently, rapidly retrain has also been applied in federated learning to tackle the unlearning problem [20, 22, 36]. Since all these approaches have to intervene the original training pipeline, they may inevitably hurt the utility of the DNN model and cannot fit the needs of practical Machine Learning as a Service (MLaaS) platforms [13, 37].

Updating Parameters for DNN Models. Another branch of researches attempt to unlearn a DNN model by updating its parameters according to the forgetting data. Golatkar *et al*. [8] propose Fisher Forgetting to scrub the

[1] https://www.dropbox.com/s/ssgzsp473aiiddl/BoundaryUnlearning-Code.zip?dl=0

weights clean of information about the forgetting data, by applying noisy Newton update on parameters of the DNN model. To locate the influence of the forgetting data at the parameter space, Fisher Forgetting adopts the Fisher Information Matrix (FIM), restricted to the remaining data, to compute the specific noise to destroy the information of the forgetting data. However, it is prohibitively expensive to compute FIM due to the large dimension of the parameter space. To alleviate this issue, Peste *et al*. [27] approximate FIM in an empirical way, which requires a single gradient computation per sample. Considering that the null-space of weights with similar activations may disable the added noise when it destroys the information of the forgetting data, Golatkar *et al*. [9] propose NTK (Neural Tangent Kernels) forgetting [17] by transforming the trained DNN model into a linearized NTK model and adding the same noise to a linearized version of the trained model. Nevertheless, the computational complexity of these methods is not that scalable as the increasing of data size. Also, we find the specific noise of these methods may still hurt the utility of the remaining data. One potential reason would be that the influence of the forgetting data at the parameter space cannot be estimated exactly due to the poor interpretability of DNNs.

3. Preliminaries and Notation

We define the notion of machine unlearning in the context of supervised classification with DNNs. Let $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{N} \subseteq \mathcal{X} \times \mathcal{Y}$ be the training dataset where $\mathbf{x}_i \in \mathcal{X}$ denotes one input and $\mathbf{y}_i \in \mathcal{Y}$ denotes the corresponding class label. $\mathcal{Y} = \{1, ..., K\}$ denotes the label space where K is the total number of classes. We denote $\mathcal{D}_f \subseteq \mathcal{D}$ as a subset of training data as the forgetting data, and $\mathcal{D}_r = \mathcal{D} \setminus \mathcal{D}_f$ as the remaining training data containing the information we expect to retain. In this work, we primarily focus on the case where $\mathcal{D}_f$ consists of the samples of an entire class.

Let the original DNN model trained on $\mathcal{D}$ be represented by $f_{\mathbf{w}_0}$ parameterized by $\mathbf{w}_0$. Given an input $\mathbf{x}$, $f_{\mathbf{w}_0}(\mathbf{x})$ is the logit output by the trained model on the input $\mathbf{x}$. Given $f_{\mathbf{w}_0}$, we aim to unlearn the information of $\mathcal{D}_f \subseteq \mathcal{D}$ from $f_{\mathbf{w}_0}$ by updating the parameters $\mathbf{w}_0 \rightarrow \mathbf{w}'$, where $\mathbf{w}'$ represents the updated parameters obtained by unlearning methods. Note that in the problem of unlearning, we expect that the unlearned model $f_{\mathbf{w}'}$ is as similar to the retrained model $f_{\mathbf{w}^*}$ as possible, and $f_{\mathbf{w}^*}$ is retrained on the remaining data $\mathcal{D}_r$ as the optimal unlearning model.

In Boundary Unlearning, we mainly focus on decision boundary of each class pair (i, j), which can be defined as: $\mathcal{B}^{(i,j)} \triangleq \{\mathbf{x} | f^i(\mathbf{x}) = f^j(\mathbf{x}) = \max_k f^k(\mathbf{x})\}$, where $i, j \in \{1, , ..., K\}$ denote labels of the class pair and $f^i(\mathbf{x})$ denotes the i-th element of the output of DNN on input $\mathbf{x}$. Let $\mathbf{x}_f \in \mathcal{D}_f$ denote a forgetting sample and the label of the forgetting class is t. Then, the prediction of $\mathbf{x}_f$ on the retrained model $f_{\mathbf{w}^*}$ will behave as $\arg\max_k f^k_{\mathbf{w}^*}(\mathbf{x}_f) \neq t$.

4. Proposed Methods

Observing that imitating the decision behavior of the retrained model $f_{\mathbf{w}^*}$ is able to accomplish the utility and privacy guarantees of machine unlearning, we transfer our attention from parameter space to decision boundary and design two strategies to shift decision boundary in Boundary Unlearning as follows.

Boundary Shrink splits the decision space of the forgetting class by the constraint: $\arg\max_k f^k_{\mathbf{w}'}(\mathbf{x}_f) = y_{nbi}$.

Boundary Expanding disperses the activation about the forgetting data by the constraint: $f^t_{\mathbf{w}'}(\mathbf{x}_f) \approx 0$.

4.1. Boundary Shrink

An intuitive boundary shifting method is to finetune the trained DNN model $f_{\mathbf{w}_0}$ with randomly labeled forgetting data, but this will also shift the boundary of the remaining class randomly, leading to the degeneration of utility the on remaining data. As can be observed from Figure 1, most of the forgetting samples are predicted as specific classes, instead of random classes. It reminds us to discover the similarity between forgetting samples and samples of other classes in the feature space.

Motivated by recent advances of adversarial attacks [29, 39], which can generate adversarial examples across the nearest decision boundary, we thus propose a neighbor searching method to identify the nearest but incorrect class labels to guide the way of boundary shifting (c.f Figure 2). Our neighbor searching shows us the direction of shifting the decision boundary of forgetting samples, and can attain the nearest but incorrect labels by predicting *the cross samples* on original model. Then we can assign the labels of the cross samples to their corresponding forgetting samples. In this way, finetuning the trained model $f_{\mathbf{w}_0}$ with all reassigned samples will shrink the boundary of the forgetting class in a precise direction.

More formally, we assume the original model $f_{\mathbf{w}_0}$ is optimized by the loss function $\mathcal{L}$, where $\mathcal{L}$ can be any standard loss functions, such as cross-entropy. $\mathbf{w}_0$ is the optimal solution of the original model:

$$\mathbf{w}_0 = \arg\min_{\mathbf{w}} \sum_{(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}} \mathcal{L}(\mathbf{x}_i, \mathbf{y}_i, \mathbf{w}) \tag{1}$$

Then, we find the nearest but incorrect label for each forgetting sample with our neighbor searching method, by adding noise vector whose elements are equal to the sign of the elements of the gradient of the loss function. Note that this is somewhat similar to adversarial attacks, but the difference is that we do not need to seek the imperceptible noise (actually, we set a relatively large noise bound),

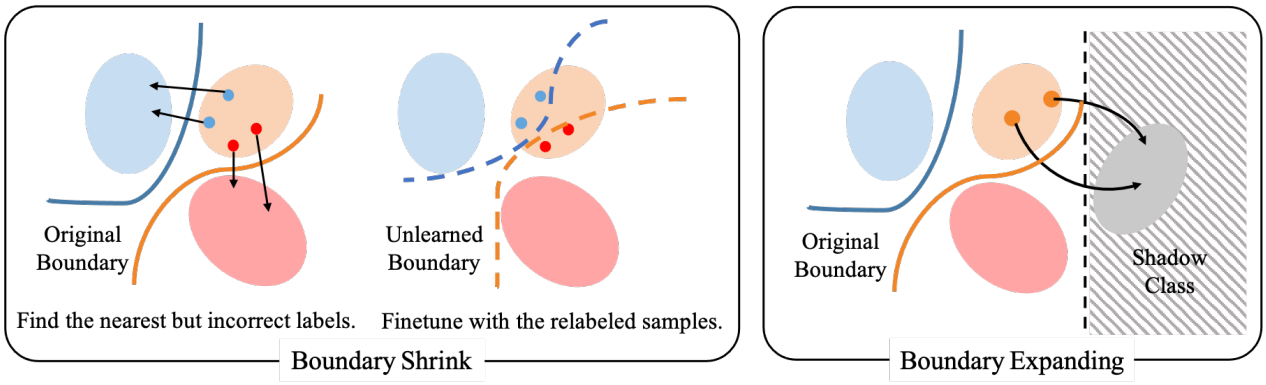


Figure 2. Illustrating the key ideas of Boundary Shrink and Boundary Expanding. The ovals in different colors represent features of samples in different categories and the dots represent the training samples of the forgetting class. Both Boundary Shrink and Boundary Expanding can destroy the decision boundary of the forgetting class.

so our method will be much faster (see Appendix for more experimental results).

Given an initial forgetting sample $\mathbf{x}_f$ and a noise bound ϵ, our neighbor searching updates its cross example using:

$$\mathbf{x'}_f = \mathbf{x}_f + \epsilon \cdot \text{sign}(\bigtriangledown_{\mathbf{x}_f} \mathcal{L}(\mathbf{x}_f, \mathbf{y}, \mathbf{w}_0)) \tag{2}$$

Once we get the cross samples of all forgetting samples, the nearest but incorrect labels $\mathbf{y}_{nbi}$ can be predicted by the original model $f_{\mathbf{w}_0}$ by:

$$\mathbf{y}_{nbi} \leftarrow \text{softmax}(f_{\mathbf{w}_0}(\mathbf{x'}_f)) \tag{3}$$

To shrink the boundary of the forgetting class, we then finetune the original model $f_{\mathbf{w}_0}$ with the reassigned forgetting samples $(\mathbf{x}, \mathbf{y}_{nbi}) \in \mathcal{D}_f$, and obtain:

$$\mathbf{w}' = \arg\min_{\mathbf{w}} \sum_{(\mathbf{x}_i, \mathbf{y}_{nbi}) \in \mathcal{D}_f} \mathcal{L}(\mathbf{x}_i, \mathbf{y}_{nbi}, \mathbf{w}_0) \tag{4}$$

For the utility guarantee, Boundary Shrink deactivates the power of the DNN model on the forgetting class, but barely hurts the generalization performance on remaining classes. The nearest but incorrect labels help to shrink the boundary of forgetting sample from the sides of other classes, which split the decision space of the forgetting class. Compare to finetuning with random labels, Boundary Shrink achieves the privacy guarantee better as well. Finetuning with random labels will make most of the forgetting samples too conspicuous, which expresses as being predicted too much confidently. By contrast, Boundary Shrink pushes the forgetting samples close to the new decision boundary, which makes the unlearned model predict on these forgetting samples with low certainty. The new boundary between remaining classes and forgetting class can be formulated as $\mathcal{B}^{(i,t)} = \{\mathbf{x}_f | \arg\max_k f^k_{\mathbf{w}'}(\mathbf{x}_f) = y_{nbi} = i\}$. Hence, attackers cannot inference these forgetting samples from the unlearned model.

4.2. Boundary Expanding

Although Boundary Shrink achieves both utility and privacy guarantees well, it may cost some time during the neighbor searching. So we propose Boundary Expanding which aims to imitate the decision behavior of the retrained model even more quickly.

In Boundary Expanding, we do not find the nearest but incorrect way to shift the boundary in decision space. Instead, we artificially assign forgetting samples to an extra shadow class of the original model, which will exploit a new area on decision space (c.f. Figure 2). Recall that we start from our observation that most of forgetting samples move to the border of other clusters on the retrained model. This phenomenon means the forgetting data are predicted as other classes with low certainty. As for a single sample, low certainty represents the output vector is more even. Based on this phenomenon, we design a boundary expanding and remapping method to disperse the activation about forgetting data of the unlearned model.

More specifically, we first introduce an extra shadow class to expand the decision space. This is done by adding a neuron at the last layer of the original DNN model. Then, we finetune the expanded model with the forgetting samples reassigned with the shadow class label:

$$\mathbf{w}' = \arg\min_{\mathbf{w}} \sum_{(\mathbf{x}_i, \mathbf{y}_{shadow}) \in \mathcal{D}_f} \mathcal{L}(\mathbf{x}_i, \mathbf{y}_{shadow}, \mathbf{w}_0) \tag{5}$$

This finetuning approach will remap the forgetting samples to the new area of decision space. After that, the activation about forgetting data will be collected in the new neuron and the classification neuron of forgetting class will be deactivated. Then, we discard this new area by pruning the extra neuron. Thus, the new model unlearned by Boundary Expanding will get back to the same size as the original model, and the pruned model will throw away the

information about the forgetting data.

Intuitively, the expanding and remapping operations only move the forgetting samples in the decision space. As Boundary Expanding never assigns any samples to the remaining classes, the activation of the remaining classes will not change much. Thus, the utility of the DNN model to the remaining data is maintained and the utility guarantee can be achieved by Boundary Expanding. Moreover, the classification neuron of the forgetting class in the pruned DNN model will be disabled. Also, the activation about forgetting data will be dispersed to the neurons belong to other classes. Therefore, attackers will only get fuzzy logits of the forgetting samples when they attack the pruned model.

5. Performance Evaluation

5.1. Experimental Settings

Datasets. We conduct experiments on CIFAR-10 [19] and Vggface2 [3] datasets, as in [8,9,34], to test unlearning performance on image classification task and face recognition task.

Baselines. We implement the following baseline unlearning methods for comparisons:

Retrain: we train the model from scratch with the remaining data as the retrained model. Thus, the retrained DNN model is the optimal unlearned model.

Finetune: we finetune the original model on the remaining training data $\mathcal{D}_r$ with large learning rate.

Random Labels [14]: finetune the original model on the random relabeled forgetting data $\mathcal{D}_f$.

Negative Gradient [8]: finetune the original model on the forgetting data $\mathcal{D}_f$ in the direction of gradient ascent.

Fisher Forgetting [8]: Fisher Forgetting first locates the most relevant parameters in terms of forgetting data and then scrub them by adding noise.

Amnesiac Unlearning [11]: amnesiac unlearning is a typical method of rapidly retrain. We need save the updates of parameters during several batches on the original training phase. Then, we can unlearn the forgetting data by subtracting the corresponding updates of parameters.

Implementations. Our methods and other baselines are implemented in Python 3.8 and use the PyTorch library [25]. All experiments are conducted on a workstation with one NVIDIA GeForce RTX 2070 GPU. For CIFAR-10 dataset, we train the All-CNN model [33] from scratch for 30 epochs using SGD with a fixed learning rate of 0.1, momentum of 0.9 and batch-size of 64. For Vggface2 dataset, we randomly select face images of 10 celebrities to conduct experiments. We obtain the original model by finetuning the pretrained ResNet50 model [3, 15] with the training images of the 10 celebrities. The original training parameters are similar to those on CIFAR-10. For the fine-tune process in Boundary Unlearning we use a learning rate of 10^{-5} for 10 epochs. Note that we use the same forgetting class as the forgetting data for all unlearning methods and the rest of classes are the remaining data.

Metrics. We verify the unlearning performance on both utility and privacy guarantees.

For utility guarantee, we utilize four accuracy metrics: *accuracy on the remaining training data* $\mathcal{D}_r$, *forgetting training data* $\mathcal{D}_f$, *remaining testing data* $\mathcal{D}_{rt}$, *forgetting testing data* $\mathcal{D}_{ft}$. The unlearned model is expected to get close accuracy with the retrained model.

For privacy guarantee, we construct a simple yet general membership inference attack (MIA) based on [32] using the output of the unlearned model and test the attack success rate (ASR). MIA aims to infer whether a data record was used to train a target ML model or not [16, 21, 28], and its ASR is widely used as an evaluation metric to verify the privacy guarantee of an unlearning method [1,5,23,30]. Ideally, a forgetting procedure should have the same attack success as a retrained model.

5.2. Utility Guarantee

For an effective unlearning method, the unlearned model is expected to contain little information about the forgetting data. So we first present comparison results of the accuracy of models by different unlearning methods on CIFAR-10 and Vggface2 datasets in Table 1, which demonstrate the difference of utility caused by the information of the forgetting data. From the results we can observe that all the baselines can erase the information of $\mathcal{D}_f$ to some degree. Finetuning the original model on $\mathcal{D}_r$ with lager learning rate and epoch significantly decreases the accuracy on $\mathcal{D}_f$ (0.0% from the initial 98.57% on Vggface2). But the more finetune epoch cost more time, and only finetuning on $\mathcal{D}_r$ fails on pursuing the privacy goal, which suggests that finetuning is not a correct unlearning method (as will be shown in the next section). Negative Gradient and Random Lables perform well on accuracy of $\mathcal{D}_r$ and $\mathcal{D}_{rt}$, which means the information of $\mathcal{D}_r$ is maintained, but they fail to erase $\mathcal{D}_f$ completely (still preserve 10.4% and 7.84% accuracy of $\mathcal{D}_f$ on CIFAR-10). In brief, these baselines unlearn the information about $\mathcal{D}_f$ insufficiently.

On the contrary, our methods achieve the utility guarantee efficiently. Owing to the well-organized moving direction, Boundary Shrink unlearns the forgetting class and maintains the information of the remaining classes in a more fine-grained way. As shown in Table 1, Boundary Shrink reduces the accuracy on $\mathcal{D}_f$ (only preserves 5.94% and 1.54% on CIFAR-10 and Vggface2) but maintains the accuracy on $\mathcal{D}_r$ (degrades 0.73% and 1.37% on the two datasets). Compared with baselines, Boundary Shrink harms the information of $\mathcal{D}_r$ to the least extent and erases $\mathcal{D}_f$ most completely. Similarly, Boundary Expanding also maintains the accuracy on $\mathcal{D}_r$ and $\mathcal{D}_{rt}$. However, it leaves more residual informa-

Table 1. Comparison of utility guarantee among baselines and Boundary Unlearning.

Dataset	Metric	Original Model	Retrain	Finetune	Negative Gradient	Random Labels	Boundary Shrink	Boundary Expanding
CIFAR-10	Acc on $\mathcal{D}_r$	99.97	100.00	100.00	97.16	98.49	**99.24**	98.03
	Acc on $\mathcal{D}_f$	99.92	0.00	0.22	7.84	10.40	**5.94**	8.96
	Acc on $\mathcal{D}_{rt}$	84.83	85.74	86.50	80.42	81.81	**83.13**	81.07
	Acc on $\mathcal{D}_{ft}$	81.20	0.00	0.10	6.50	7.50	**5.94**	7.00
Vggface2	Acc on $\mathcal{D}_r$	99.94	100.00	99.52	96.57	**98.89**	98.57	98.20
	Acc on $\mathcal{D}_f$	98.57	0.00	0.00	2.85	4.29	**1.54**	4.22
	Acc on $\mathcal{D}_{rt}$	98.87	99.06	99.96	99.58	95.14	**99.72**	97.12
	Acc on $\mathcal{D}_{ft}$	97.14	0.00	5.52	7.26	2.86	**0.87**	1.41

Table 2. Comparison of utility guarantee among Fisher Forgetting and Amnesiac Unlearning on CIFAR-10.

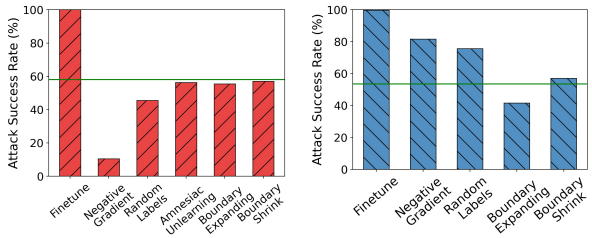
Metrics	Acc on $\mathcal{D}_r$	Acc on $\mathcal{D}_f$	Acc on $\mathcal{D}_{rt}$	Acc on $\mathcal{D}_{ft}$
Amnesiac Unlearning	95.79	0.00	81.50	0.00
Fisher Forgetting	61.62	1.80	54.20	1.60

tion about $\mathcal{D}_f$ than Boundary Shrink (8.96% and 5.94% on CIFAR-10), but still less than Random Labels (10.40% on CIFAR-10). As mentioned before, we take Boundary Expanding as a faster alternative method, which takes a trade-off between performance and time consumption.

We also run Fisher Forgetting on CIFAR-10, which destroys the information about $\mathcal{D}_f$ by adding a specific noise to parameters, to demonstrate the effectiveness of our methods. The results in Table 2 show that Fisher Forgetting erases $\mathcal{D}_f$ optimally (only preserves 1.8% accuracy on $\mathcal{D}_f$) but fails to maintain the utility on $\mathcal{D}_r$ (decreases 38.35% accuracy). However, both Boundary Shrink and Boundary Expanding achieve forgetting and maintain the information of $\mathcal{D}_r$ well. We also show the utility performance of Amnesiac Unlearning in Table 2. Amnesiac Unlearning erases information about $\mathcal{D}_f$ ideally, but still hurts the utility of $\mathcal{D}_r$ as it intervenes the original training process. Most importantly, Amnesiac Unlearning costs too much time and memory space to accomplish unlearning, as will be shown in the following section.

5.3. Privacy Guarantee

Next, we turn to evaluate the privacy guarantee to ensure that the unlearned model by our Boundary Unlearning does not leak any information about the forgetting data. We plot the ASR for the unlearning methods on CIFAR-10 and Vggface2 datasets, and the results are shown in Figure 3. The green line in Figure 3 represent the ASR of the retrained model and all the unlearning methods attempt to match with it: the closer the better. Here, as an unlearned model gets a closer ASR to 100%, the less information about $\mathcal{D}_f$ is



(a) On CIFAR-10. (b) On Vggface2.

Figure 3. The attack success rate of each unlearning method. The green line represents the ASR of the retrained model, and it is expected to be more close to it to reduce the privacy leakage.

erased. Contrarily, a quite low ASR (close to 0%) may represent the Streisand effect which can provide more information about the forgetting data [8], i.e. all samples of $\mathcal{D}_f$ are predicted as one class.

From the results in Figure 3, we can see that the ASR of Finetune is fairly high on both datasets, which means Finetune barely removes information about forgetting data. For Negative Gradient on CIFAR-10 dataset, it gets a pretty low ASR which is far from the ASR of the retrained model. Random Labels can erase a part of forgetting data but only achieves a relatively low ASR on CIFAR-10. Interestingly, both Negative Gradient and Random Labels have high ASR on Vggface2 dataset, mainly because Vggface2 consists of millions of face images and the basic patterns of faces are more difficult to erase, especially for weaker unlearning methods. Similar to the test of utility guarantee, we also test Fisher Forgetting, but find that the ASR of Fisher Forgetting is 0 which represents the severe Streisand effect. Amnesiac Unlearning obtains an ASR close to the retrained model, as it saves and then subtracts the updates of parameters related to $\mathcal{D}_f$, which makes it more like a rapid retraining model.

As for our Boundary Shrink/Expanding, they can achieve quite close ASR to that of the retrained model on both datasets. To be more specific, Boundary Shrink

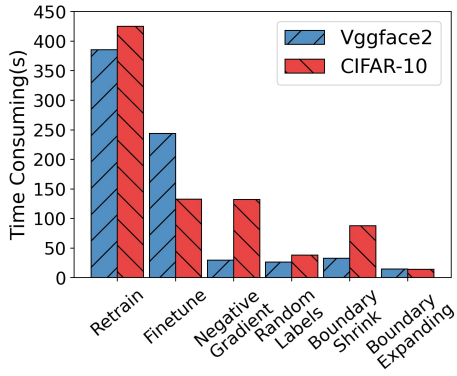


Figure 4. The time consumption of each unlearning method.

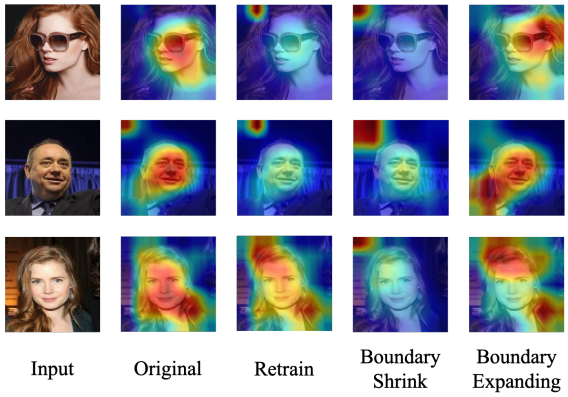


Figure 5. The attention map of each unlearning method.

achieves better ASR than Boundary Expanding because Boundary Shrink particularly considers the distance between forgetting samples and samples belong to other classes. In a word, Boundary Unlearning methods can achieve a better performance on privacy guarantee.

5.4. Computational Complexity

In this section, we report the time consumed by each unlearning method to show the computational complexity. The major results on CIFAR-10 and Vggface2 are depicted in Figure 4. Compared with Retrain, all other unlearning methods of course spent less time, but Finetune takes much more time than other methods, and it still gets a fairly high ASR in respect of privacy guarantee. As for Boundary Unlearning, both of them can forget the forgetting data in a short time and achieve the privacy guarantee well in the mean time. On the Vggface2 dataset, Boundary Expanding and Boundary Shrink reduce the forgetting time by 26.6× and 11.7×, respectively. As for CIFAR-10 dataset, Boundary Expanding and Boundary Shrink provide a speed-up of 29.7× and 4.8×, respectively. As we discussed before, Boundary Shrink takes a little more time than Boundary Expanding because the generation of cross samples.

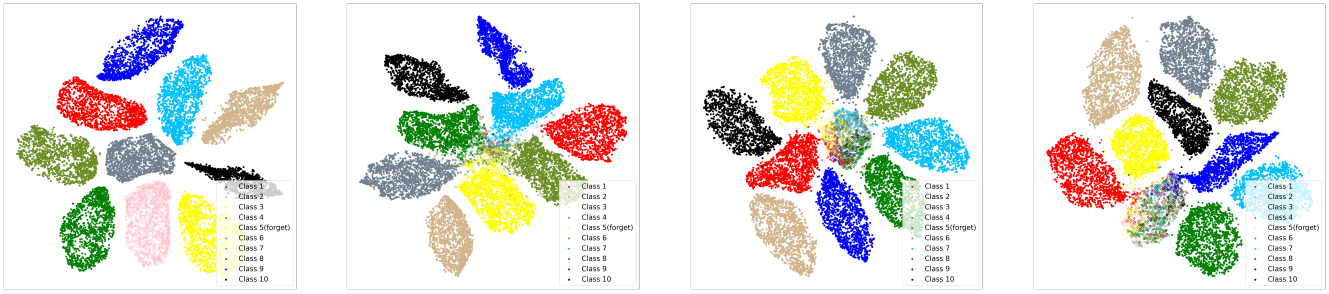
Also, we test the time consumption of Fisher Forgetting and Amnesiac Unlearning. We do not add the results to Figure 4 because both of them cost extremely long time and the results cannot fit the figure well. For CIFAR-10 on All-CNN model, Fisher Forgetting costs around 2.7×10^3 seconds to unlearn the forgetting class, mainly caused by the huge parameter space and the large amount of training samples. Although Fisher Forgetting can erase the forgetting data perfectly, the time consumption is intolerable. As for Amnesiac Unlearning, the unlearning process consists of parameters subtracting and repairing, which costs around 2.8×10^2 seconds. But from our reproduction, we find that Amnesiac Unlearning doubles the time consumption of original training process (costs around 1.0×10^3 seconds), which is mainly caused by the operation of saving parameters update. Thus, we can see that our Boundary Unlearning can remove the forgetting data effectively and quickly.

5.5. Attention Map

To make the effect of forgetting more transparent, we plot the attention maps of models on the forgetting data in the Vggface2 dataset before and after applying Boundary Unlearning. The attention map [31] highlights the important areas in the image for predicting the concept. The columns in Figure 5 (from left to right) show the focus areas of the original model, the retrained model and unlearned models generated by Boundary Unlearning. We find that the retrained model does not focus on the area of face, and Boundary Unlearning transfers the attention of models from face to background. In particular, the model unlearned by Boundary Shrink only focuses on the background. Although Boundary Expanding fails to transfer the attention completely, it still refocuses on the area outside of face. The results indicate that the output of unlearned model contains hardly any information about the forgetting data.

5.6. Visualization of Decision Space

To make more clear how the decision boundary of the forgetting data shifts, we visualize the decision boundary before and after unlearning in Figure 6. We can find that the forgetting data (hollow circles) has been predicted as the nearest classes after applying Boundary Shrink, even the cluster does not entirely spread out like on the retrained model. In addition, we find that some hollow circles move to other clusters in Figure 6c. These phenomenons reveal that the decision space of the forgetting class is split by its near classes, like on the decision space of the retrained model (c.f. Figure 6b). Meanwhile, the clusters of remaining classes still keep compact, which means the model utility to the remaining data is maintained. In the decision space of the unlearned model generated by Boundary Ex-



(a) Original (b) Retrained (c) Boundary Shrink (d) Boundary Expanding

Figure 6. Visualization of decision space of different methods on CIFAR-10 dataset. The solid dots in different colors represent samples belonging to different remaining classes and the hollow circles stand for the forgetting data. Ideally, we wish the unlearned models act like the retrained model.

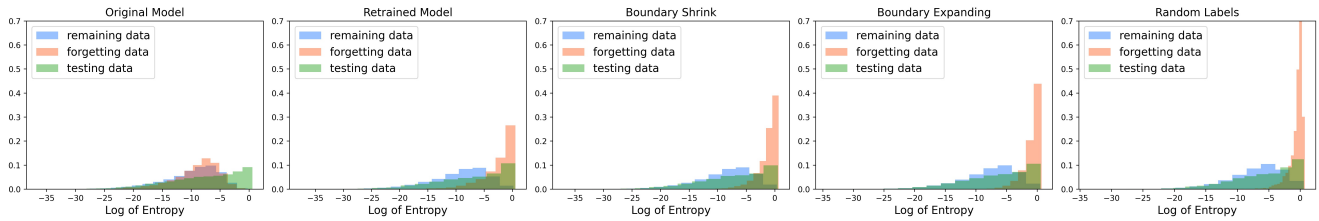


Figure 7. Distribution of the entropy of model's output on the remaining data $\mathcal{D}_r$, the forgetting data $\mathcal{D}_f$ and the testing data.

panding shown in Figure 6d, the cluster of the forgetting data is pushed away from the center. The forgetting samples are predicted as the remaining classes. Moreover, the clusters of remaining classes are maintained and few remaining data is predicted incorrectly. Therefore, Boundary Unlearning makes the decision boundary of the unlearned model more like that of the retrained model and thus accomplishes the unlearning efficacy.

5.7. Distribution of the Entropy of Model Output

At last, we deep into the distribution of the model's output to figure out why Boundary Unlearning can achieve privacy guarantee. With respect to the distribution of the model's output on the forgetting data, we expect the unlearned models to match closely with the retrained model. On the other side, a great difference between distributions before and after unlearning will give rise to Streisand effect.

Figure 7 shows the distribution of entropy of model's output. From the results, we can see that the entropy of outputs on $\mathcal{D}_r$ and $\mathcal{D}_f$ is lower (more confident) than that on the testing data, because both of them are training data of the original model. As the retrained model has not trained on $\mathcal{D}_f$, the entropy of $\mathcal{D}_f$ on it increases evidently (less confident). The distributions of unlearned models generated by Boundary Unlearning are more similar to that of the retrained model. For the distribution on the unlearned model with Random Labels, its entropy of $\mathcal{D}_f$ is pretty large and vary significantly than that of the retrained model, which may provide even more information about $\mathcal{D}_f$, i.e., all the samples of $\mathcal{D}_f$ are predicted as a specific class by the unlearned model, which may make $\mathcal{D}_f$ more prominent to attackers.

We can also observe that scrubbing $\mathcal{D}_f$ from the original model by Boundary Unlearning makes the distribution of model's output on $\mathcal{D}_f$ more uniform, which means the model is less confident about the prediction. We believe that the change about the distribution is caused by the shifting of the decision boundary. Boundary Unlearning destroys the boundary of the forgetting class and thus $\mathcal{D}_f$ is predicted as one of its nearest classes, as shown in Figures 6c and 6d, which means the unlearned model predicts them with low confidence like predicting the testing samples. This will make the attackers harder to inference the membership information about $\mathcal{D}_f$.

6. Conclusion

This paper has presented Boundary Unlearning, the first machine unlearning method to remove information of an entire class from a trained DNN by shifting the decision boundary. Boundary Unlearning neither costs too much computational resource nor intervenes the original training pipeline. From extensive experiments on different tasks, Boundary Unlearning demonstrates rapid and efficient forgetting performance in both utility and privacy guarantees of unlearning. We envision our work as a practical step in machine unlearning towards revealing the relationship be-

tween decision boundary and forgetting.

References

[1] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *Proceedings of IEEE S&P*, pages 141–159. IEEE, 2021. 1, 2, 5

[2] Jonathan Brophy and Daniel Lowd. Machine unlearning for random forests. In *Proceedings of ICML*, pages 1092–1104, 2021. 2

[3] Qiong Cao, Li Shen, Weidi Xie, Omkar M Parkhi, and Andrew Zisserman. Vggface2: A dataset for recognising faces across pose and age. In *Proceedings of IEEE International Conference on Automatic Face & Gesture Recognition*, pages 67–74, 2018. 5

[4] Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *Proceedings of IEEE S&P*, pages 463–480, 2015. 2

[5] Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, Mathias Humbert, and Yang Zhang. When machine unlearning jeopardizes privacy. In *Proceedings of ACM CCS*, pages 896–911, 2021. 5

[6] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of ACM CCS*, pages 1322–1333, 2015. 1

[7] Antonio Ginart, Melody Guan, Gregory Valiant, and James Y Zou. Making ai forget you: Data deletion in machine learning. In *Proceedings of NeurIPS*, volume 32, 2019. 2

[8] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of IEEE/CVF CVPR*, pages 9304–9312, 2020. 1, 2, 5, 6

[9] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Forgetting outside the box: Scrubbing deep networks of information accessible from input-output observations. In *Proceedings of ECCV*, pages 383–398, 2020. 1, 3, 5

[10] Micah Goldblum, Dimitris Tsipras, Chulin Xie, Xinyun Chen, Avi Schwarzschild, Dawn Song, Aleksander Madry, Bo Li, and Tom Goldstein. Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses. *on Pattern Analysis and Machine Intelligence*, to appear. DOI: 10.1109/TPAMI.2022.3162397. 1

[11] Laura Graves, Vineel Nagisetty, and Vijay Ganesh. Amnesiac machine learning. In *Proceedings of AAAI*, volume 35, pages 11516–11524, 2021. 1, 2, 5

[12] Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified data removal from machine learning models. In *Proceedings of ICML*, pages 3832–3842, 2020. 2

[13] Lucjan Hanzlik, Yang Zhang, Kathrin Grosse, Ahmed Salem, Maximilian Augustin, Michael Backes, and Mario Fritz. Mlcapsule: Guarded offline deployment of machine learning as a service. In *Proceedings of IEEE/CVF CVPR*, pages 3300–3309, 2021. 2

[14] Tomohiro Hayase, Suguru Yasutomi, and Takashi Katoh. Selective forgetting of deep networks at a finer level than samples. *CoRR, arXiv: 2012.11849*, 2020. 5

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF CVPR*, pages 770–778, 2016. 5

[16] Hongsheng Hu, Zoran Salcic, Lichao Sun, Gillian Dobbie, Philip S Yu, and Xuyun Zhang. Membership inference attacks on machine learning: A survey. *ACM Computing Surveys*, 54(11):1–37, 2022. 5

[17] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Proceedings of ICLR*, 31, 2018. 3

[18] Mahdi Khosravy, Kazuaki Nakamura, Yuki Hirose, Naoko Nitta, and Noboru Babaguchi. Model inversion attack by integration of deep generative models: Privacy-sensitive face generation from a face recognition system. *IEEE Transactions on Information Forensics and Security*, 17:357–372, 2022. 1

[19] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Master's Thesis*, 2009. 5

[20] Gaoyang Liu, Xiaoqiang Ma, Yang Yang, Chen Wang, and Jiangchuan Liu. Federaser: Enabling efficient client-level data removal from federated learning models. In *Proceedings of IEEE/ACM IWQoS*, pages 1–10, 2021. 2

[21] Gaoyang Liu, Tianlong Xu, Xiaoqiang Ma, and Chen Wang. Your model trains on my data? protecting intellectual property of training data via membership fingerprint authentication. *IEEE Transactions on Information Forensics and Security*, 17:1024–1037, 2022. 5

[22] Yi Liu, Lei Xu, Xingliang Yuan, Cong Wang, and Bo Li. The right to be forgotten in federated learning: An efficient realization with rapid retraining. In *Proceedings of IEEE INFOCOM*, 2022. 2

[23] Ronak Mehta, Sourav Pal, Vikas Singh, and Sathya N. Ravi. Deep unlearning via randomized conditionally independent hessians. In *Proceedings of IEEE/CVF CVPR*, pages 10422–10431, 2022. 1, 5

[24] Hieu T Nguyen and Arnold Smeulders. Active learning using pre-clustering. In *Proceedings of ICML*, 2004. 1

[25] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Proceedings of NeurIPS*, 32, 2019. 5

[26] Neehar Peri, Neal Gupta, W Ronny Huang, Liam Fowl, Chen Zhu, Soheil Feizi, Tom Goldstein, and John P Dickerson. Deep k-nn defense against clean-label data poisoning attacks. In *Proceedings of ECCV*, pages 55–70, 2020. 1

[27] Alexandra Peste, Dan Alistarh, and Christoph H. Lampert. SSSE: Efficiently erasing samples from trained machine learning models. *CoRR, arXiv: 2107.03860*, 2021. 1, 3

[28] Shahbaz Rezaei and Xin Liu. On the difficulty of membership inference attacks. In *Proceedings of IEEE/CVF CVPR*, pages 7892–7900, 2021. 5

[29] Ishai Rosenberg, Asaf Shabtai, Yuval Elovici, and Lior Rokach. Adversarial machine learning attacks and defense methods in the cyber security domain. *ACM Computing Surveys*, 54(5):1–36, 2021. 3

[30] Ayush Sekhari, Jayadev Acharya, Gautam Kamath, and Ananda Theertha Suresh. Remember what you want to forget: Algorithms for machine unlearning. In *Proceedings of NeurIPS*, volume 34, pages 18075–18086, 2021. 5

[31] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of IEEE/CVF ICCV*, pages 618–626, 2017. 7

[32] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *Proceedings of IEEE S&P*, pages 3–18, 2017. 5

[33] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *CoRR, arXiv: 1412.6806*, 2014. 5

[34] Ayush K Tarun, Vikram S Chundawat, Murari Mandal, and Mohan Kankanhalli. Fast yet effective machine unlearning. *CoRR, arXiv: 2111.08947*, 2021. 5

[35] Paul Voigt and Axel Bussche. *The EU General Data Protection Regulation (GDPR): A Practical Guide*. Springer, 2017. 1

[36] Junxiao Wang, Song Guo, Xin Xie, and Heng Qi. Federated unlearning via class-discriminative pruning. In *Proceedings of ACM Web Conference*, pages 622–632, 2022. 2

[37] Qizhen Weng, Wencong Xiao, Yinghao Yu, Wei Wang, Cheng Wang, Jian He, Yong Li, Liping Zhang, Wei Lin, and Yu Ding. MLaaS in the wild: Workload analysis and scheduling in large-scale heterogeneous GPU clusters. In *Proceedings of USENIX NSDI*, pages 945–960, 2022. 2

[38] Yinjun Wu, Edgar Dobriban, and Susan Davidson. Deltagrad: Rapid retraining of machine learning models. In *Proceedings of ICML*, pages 10355–10366, 2020. 1, 2

[39] Han Xu, Yaxin Li, Wei Jin, and Jiliang Tang. Adversarial attacks and defenses: Frontiers, advances and practice. In *Proceedings of ACM KDD*, pages 3541–3542, 2020. 3