This is the peer reviewd version of the followng article:

Unlearning Vision Transformers without Retaining Data via Low-Rank Decompositions / Poppi, Samuele; Sarto, Sara; Cornia, Marcella; Baraldi, Lorenzo; Cucchiara, Rita. - (2024). (Intervento presentato al convegno 27th International Conference on Pattern Recognition tenutosi a Kolkata, India nel December 01-05, 2024).

Terms of use:

The terms and conditions for the reuse of this version of the manuscript are specified in the publishing policy. For all terms of use and more information see the publisher's website.

21/12/2024 17:30

(Article begins on next page)

Unlearning Vision Transformers without Retaining Data via Low-Rank Decompositions

Samuele Poppi[1,2], Sara Sarto[1], Marcella Cornia[1], Lorenzo Baraldi[1], and Rita Cucchiara[1]

[1] University of Modena and Reggio Emilia, Italy
{name.surname}@unimore.it
[2] University of Pisa, Italy
name.surname@phd.unipi.it

Abstract. The implementation of data protection regulations such as the GDPR and the California Consumer Privacy Act has sparked a growing interest in removing sensitive information from pre-trained models without requiring retraining from scratch, all while maintaining predictive performance on remaining data. Recent studies on machine unlearning for deep neural networks have resulted in different attempts that put constraints on the training procedure and which are limited to small-scale architectures and with poor adaptability to real-world requirements. In this paper, we develop an approach to delete information on a class from a pre-trained model, by injecting a trainable low-rank decomposition into the network parameters, and without requiring access to the original training set. Our approach greatly reduces the number of parameters to train as well as time and memory requirements. This allows a painless application to real-life settings where the entire training set is unavailable, and compliance with the requirement of time-bound deletion. We conduct experiments on various Vision Transformer architectures for class forgetting. Extensive empirical analyses demonstrate that our proposed method is efficient, safe to apply, and effective in removing learned information while maintaining accuracy.

Keywords: Machine Unlearning · Low-Rank Adaptation · Vision Transformers · Image Classification.

1 Introduction

Unlearning, the task of removing the impact of specific training data from a pre-trained model [35], is gaining attention in the Machine Learning community due to data protection laws like GDPR and the California Consumer Privacy Act [18,43], that aim to guarantee every user with the "right to be forgotten" [14, 15] and require companies to erase personal data and their impact on trained models, upon request. Solutions aim to erase the influence of certain data points, essentially "untraining" the model to resemble one trained without them [4,7,24].

While removing one or more specific datapoints is crucial for addressing privacy concerns, the literature has also been recently investigating the removal

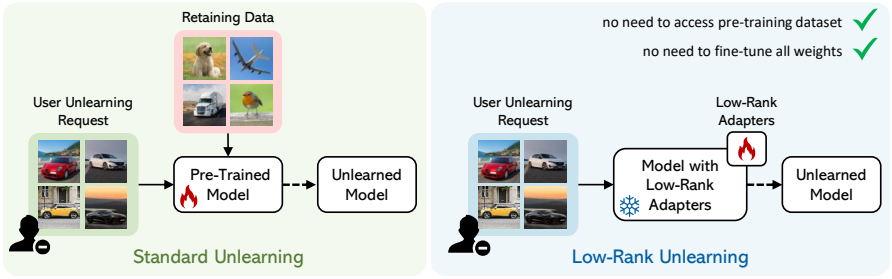

Fig. 1. Overview of our approach: we unlearn a class from a pre-trained image classification network without requiring access to the pre-training dataset and by injecting a low-rank adaptation matrix into the network weights.

of entire classes from a classification network [2, 33, 34, 39]. This setting, which is much more complex in both computational and algorithmic terms, is needed when the entire class is a source of privacy leak (*e.g.*, in a face recognition system), but also opens up new possibilities in terms of removing portions of the knowledge learned by the model when it is detrimental or not needed for a specific application scenario.

Previous attempts in this direction have tackled the unlearning phase as a fine-tuning step that involves all weights of a neural network [4, 20, 42], which makes untraining computationally complex and hardly feasible in practical scenarios in which a neural network, that is employed in production needs to be quickly adapted in response to a data removal request. Moreover, most approaches to unlearning require the load and access of the entire training dataset over which the network has been trained [19, 20, 42], putting another constraint on the practical applicability of previous approaches in a production environment. In these scenarios, indeed, the pre-training dataset can be significantly large or even not available if the model has been trained on private data and acquired from a third party.

To address these issues, we propose an unlearning algorithm that does not require fine-tuning the entire set of parameters of a pre-trained model and that does not need access to the dataset employed at training time (see Fig. 1). For each layer of a pre-trained neural network, we learn a low-rank matrix that is summed to the pre-trained parameters of the layer, with the aim of both forgetting unwanted data and retaining the original knowledge of the network. This is achieved by modeling a low-rank trainable decomposition and leaving the rest of the layers frozen, significantly reducing the number of trainable parameters and optimizer states needed for untraining. At the same time, this solution increases the network retaining capability and limits the loss in performance on the portion of the data to preserve, thus addressing one of the most important challenges in unlearning [12]. What is more, differently from many recent proposals [19, 42], this approach does not require explicit knowledge of the pre-training dataset. Indeed, the objective of retaining the original knowledge is achieved by regularizing the low-rank decomposition to increase its sparsity. Therefore, our

approach can be applied without storing the pre-training dataset, thus lowering the storage requirements and allowing unlearning on backbones whose training dataset is not known or available.

Experimentally, we validate the proposed approach on modern image classification architectures based on the Vision Transformer [17] on CIFAR-10, CIFAR-20, and ImageNet-1k datasets, demonstrating its applicability and efficiency. In summary, our major contributions are as follows:

- We propose low-rank unlearning, the first approach to unlearn an entire class from a neural network by injecting a low-rank decomposition into the network parameters. Compared to existing work, low-rank unlearning greatly reduces the amount of computational resources required for unlearning.
- Our method is based on learning low-rank trainable matrices. In contrast to previous works which require complete access to retaining data, our approach allows for modeling an unsupervised retaining objective that does not require loading the pre-training dataset.
- We conduct extensive experiments to evaluate low-rank unlearning on image classification tasks. The results show that low-rank unlearning can rapidly and effectively forget undesired classes and outperform existing techniques.

2 Related Work

Machine unlearning. Research efforts initially focused on unlearning solutions for traditional machine learning algorithms [7]. Instead, more recent approaches targeted erasing specific data points or entire classes from pre-trained deep neural networks [4, 20, 28, 29, 42]. One approach involves retraining the model from scratch on the remaining data, which is computationally demanding, especially for large-scale deep neural networks. To address these limitations, some works aim to accelerate the retraining process [4,10,21,44]. In this context, Bourtoule *et al.* [4] suggested partitioning the retraining dataset into shards to minimize data requirements. Another solution [21] involves storing and reusing gradient information during training. However, these methods often require modifying the original training process and are not easily applicable in real-world scenarios.

A different research line has focused on developing effective strategies to update network parameters according to the samples the model should forget, without retraining the entire model from scratch [11, 20, 42]. For example, Golatkar *et al.* [20] introduced a scrubbing procedure to remove information from parameters by adding noise. However, these methods face challenges with large datasets. To address this issue, a subsequent work [19] proposed splitting the network weights into core non-linear weights and linear user parameters, allowing for selective deletion without loss of accuracy.

Recently, other proposals for tackling the unlearning task have been presented, where unlearning is done either by retraining the model with a teacher-student paradigm with competent and incompetent teachers [42] or by shifting the decision boundary of a deep neural network to emulate the behavior of a model trained without samples of the forget class [9]. While almost all

the above-mentioned works need the set of data the network should not forget, Cha *et al.* [8] proposed a novel instance-wise unlearning framework in which a set of instances is deleted from the original model by intentionally misclassifying them. Our work, in contrast, focuses on unlearning entire classes through a fine-tuning strategy that involves learning low-rank decomposition matrices. This allows for a reduction of the computational complexity of the unlearning procedure while achieving good retaining capabilities.

Some other research efforts have been dedicated to few-shot [36,45] and zero-shot [12] machine unlearning. While the former concerns a setting in which only a few samples of the target data are available, the latter imposes the constraint that no training data are available to perform the unlearning task. While these settings are closely related to our proposal, we instead place ourselves in a more realistic scenario, in which all the unwanted samples are available, while having access to the rest of the pre-training dataset is not required.

Low-rank adaptation. During fine-tuning, updating all parameters of a pre-trained model is computationally expensive. Parameter-Efficient Fine-Tuning (PEFT) addresses this problem by optimizing a small portion of parameters, leaving the backbone model unchanged. Among PEFT methods, Low-Rank Adaptation (LoRA) [23] is one of the most popular approaches since it only requires tuning small low-rank matrices, achieving comparable performance compared to full fine-tuning across a wide range of tasks. For its efficiency, LoRA has been used across different research fields, like the fine-tuning of large language models to adapt them for various multimodal tasks [5,6] and foundation models fine-tuning for improving their safety [38]. LoRA has also been used in facing the severe risk of privacy leakage in latent diffusion models [31]. Moreover, in federated learning, LoRA can be used to update local models efficiently without sharing the full model parameters. This ensures that sensitive data remains on local devices, enhancing privacy [41]. In contrast to previous research, we are the first, to the best of our knowledge, to design a LoRA-based solution to unlearn classes from pre-trained classification backbones.

3 Proposed Method

3.1 Preliminaries

Notation. Let $\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N} \subseteq \mathcal{X} \times \mathcal{Y}$ be the complete training set over which a deep neural network classification model has been trained, where $\boldsymbol{x}_i \in \mathcal{X}$ denotes an input image and $\boldsymbol{y}_i \in \mathcal{Y} = \{1, ..., K\}$ its corresponding label, being K the number of classes. We denote with $\mathcal{D}_f \subset \mathcal{D}_{\text{train}}$ a set of training items whose impact needs to be removed from the model (*i.e.* the *forget set*) and $\mathcal{D}_r = \mathcal{D}_{\text{train}} \setminus \mathcal{D}_f$ the remaining set of training items over which we want to keep prediction accuracy (more formally, the *retaining set*). In this work, we focus on the *class unlearning* scenario, in which $\mathcal{D}_f$ will consist of all items from one class. Also, let $\mathcal{D}_{\text{test}}$ denote the test set used for evaluation.

Further, let $g_{\theta_0} : \mathcal{X} \to \mathcal{Y}$ indicate the original classification model pre-trained on $\mathcal{D}_{\text{train}}$, parametrized by a set of parameters θ_0. The objective of the unlearning

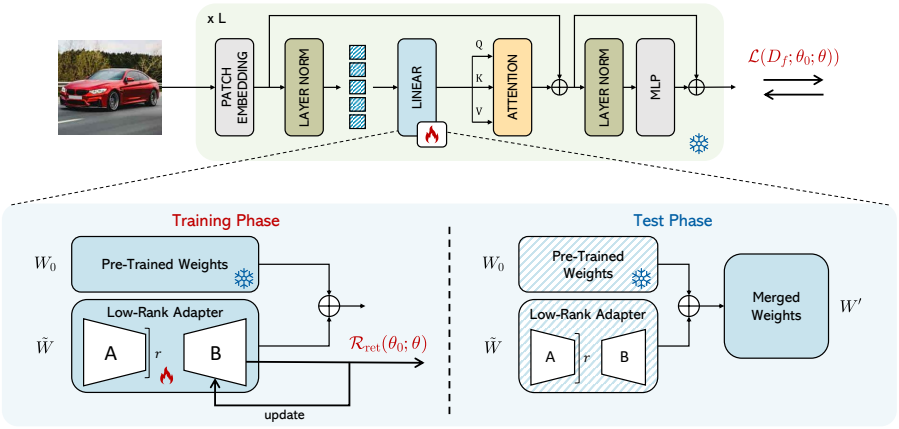

Fig. 2. Overview of the proposed low-rank unlearning solution. The pre-trained model is endowed with a trainable low-rank adapter, which is summed to existing network weights. During test, the low-rank adaptation is accumulated in the pre-trained weights to disable access to the previous state of the network.

phase is to fine-tune g by moving θ towards a state of the parameters θ' where the information of $\mathcal{D}_f$ is unlearned and the information in $\mathcal{D}_r$ is maintained. The resulting model, therefore, should behave similarly to a model g_{θ^*} which has been trained from scratch on $\mathcal{D}_r$ and which has never received gradient from samples in $\mathcal{D}_f$.

No-retain unlearning. Differently from previous works which require access to both $\mathcal{D}_f$ and $\mathcal{D}_r$ (or none of them, such as in the *zero-shot* setting defined in [12]), we suppose to have access to the pre-trained model g_{θ_0} and the forget set $\mathcal{D}_f$, without requiring access to the larger retain set $\mathcal{D}_r$. Our setting is more grounded and practical than previous ones. Indeed, it is reasonable to hypothesize that the data holder has natural access to the data that needs to be removed, *i.e.* $\mathcal{D}_f$. Also, the right to be forgotten gives the data holder up to a month to remove the data, which largely settles our approach within the bounds permitted by law. On the other hand, it is desirable that an unlearning method does not need to process the whole training set, *i.e.* $\mathcal{D}_r$. The original training set, indeed, might not be completely known or available, as in the case of models pre-trained on private data and then fine-tuned. Even when the full training set is available, however, processing during the unlearning phase inevitably requires higher computational costs and also increases storage costs.

3.2 Class-wise unlearning

The objective of unlearning is that of removing the impact of the set of samples in $\mathcal{D}_f$. This can be achieved by either imposing a misclassification or a re-labeling of those datapoints. In the first case, the network is trained to misclassify all datapoints in $\mathcal{D}_f$, *i.e.* so that $g_{\theta'}(\boldsymbol{x}) \neq \boldsymbol{y}$ for $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{D}_f$. This can be done by

performing gradient ascent over a classification loss with ground-truth labels, computed over the forget set, *i.e.*

$$\mathcal{L}_{\text{unl}}(\mathcal{D}_f;\theta) = -\mathbb{E}_{\boldsymbol{x},\boldsymbol{y}\in\mathcal{D}_f}\mathcal{L}_{\text{CE}}(g_{\theta'}(\boldsymbol{x}),\boldsymbol{y};\theta). \tag{1}$$

In the second case, instead, a gradient descent learning can be performed over a classification loss with labels different from the ground-truth ones for all the samples in $\mathcal{D}_f$, *i.e.*

$$\mathcal{L}_{\text{unl}}(\mathcal{D}_f;\theta) = \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}\in\mathcal{D}_f}\mathcal{L}_{\text{CE}}(g_{\theta'}(\boldsymbol{x}),\boldsymbol{y}';\theta), \text{ with } \boldsymbol{y}' \neq \boldsymbol{y}. \tag{2}$$

Whatever the above choice is, one of these unlearning objectives alone would induce forgetting what the network has learned on $\mathcal{D}_r$, therefore reducing its final performance after responding to a removal request. Under a setting in which direct access to $\mathcal{D}_r$ is allowed, this could be avoided by performing gradient descent with a cross-entropy loss on $\mathcal{D}_r$, to refresh its knowledge continuously during untraining. In our scenario, in which access to $\mathcal{D}_r$ is not considered, instead, a regularization loss can be added to keep the weights of the network close to θ_0 during the unlearning phase. The unlearning loss, therefore, becomes

$$\mathcal{L}(\mathcal{D}_f,\theta_0;\theta) = \mathcal{L}_{\text{unl}}(\mathcal{D}_f;\theta) + \mathcal{R}_{\text{ret}}(\theta_0;\theta), \tag{3}$$

where $\mathcal{R}(\cdot)$ is a regularizer that aims at overcoming forgetting of knowledge on the remaining data $\mathcal{D}_r$. Usually, this regularization is implemented by considering either the magnitude of weight change (*i.e.* $\theta' - \theta_0$) and its sparsity [25] or sample importance [8].

3.3 Low-rank unlearning

The unlearning procedure is, essentially, a fine-tuning process $\theta_0 \rightarrow \theta'$ induced by the loss $\mathcal{L}$. While previous literature has focused on fine-tuning the entire set of parameters θ_0 without imposing constraints on the selection of trainable weights, we instead hypothesize that this fine-tuning should happen in a low-rank space. Under this hypothesis, a complete fine-tuning of θ_0 is unnecessary and, potentially, also detrimental as it leaves the door open to overfitting on $\mathcal{D}_{\text{train}}$. In a scenario in which $\mathcal{D}_r$ is not accessible, moreover, constraining the unlearning phase to happen in a low-rank space helps to retain the original knowledge of the model that has been learned on $\mathcal{D}_r$.

Without loss of generality, in the following, we describe our approach for the case of a fully-connected layer. While these are a key ingredient of many Transformer-based models as they build up the attention operator, our approach can also straightforwardly be extended to convolutional layers. Given a pre-trained layer f, with weight $W_0 \in \theta_0$, $W_0 \in \mathbb{R}^{d\times k}$ and bias $b \in \theta_0$, which applies a transformation $f(x) = xW_0^\intercal + b$ to its input tensor $x \in \mathbb{R}^k$, we re-parametrize its transformation during the unlearning phase by adding a low-rank trainable component $\tilde{W}$, initialized from zero. We then fine-tune only the

low-rank decomposition, leaving the rest of the layer frozen. Formally,

$$f(x) = x \underbrace{W_0^\intercal}_{❄} + x \overbrace{\tilde{W}^\intercal}^{\bowtie} + \underbrace{b}_{❄} \text{ . with } \tilde{W} = BA, \tag{4}$$

where A and B provide a bottleneck that creates a low-rank decomposition (denoted with $\bowtie$, above), with $B \in \mathbb{R}^{d \times r}$, $A \in \mathbb{R}^{r \times k}$ and r being the rank of the decomposition. During unlearning, W_0 and b are kept frozen (❄) and we backpropagate gradient only on A and B. These are respectively initialized with a Gaussian initialization and with zero, so that, at the beginning of the unlearning phase $\tilde{W} = BA$ is a zero matrix and f behaves exactly as in the pre-trained state.

Constraining the unlearning phase inside the low-rank decomposition $\bowtie$ also provides a straightforward way to overcome the forgetting of knowledge with respect to $\mathcal{D}_r$, as limiting the magnitude of weight change during the unlearning phase can be done by simply constraining the magnitude of $\tilde{W}$. In continuity with previous works that suggest that unlearning should produce a sparse update of weights, we constrain $\tilde{W}$ to be sparse by adding an L_1 regularization on B, as follows:

$$\mathcal{R}_{\text{ret}}(\theta_0; \theta) = \lambda \|\text{vec}(B)\|_1, \tag{5}$$

where $\text{vec}(\cdot)$ is the vectorization operator and λ a scalar non-trainable constant. As it can be noticed, this induces B to be sparse, which in turn makes $\tilde{W}$ sparse. The regularizer can then be plugged into any unlearning loss, to realize an unlearning procedure that does not require access to the retain set. In the case of unlearning via misclassification, the complete loss thus becomes

$$\mathcal{L}(\mathcal{D}_f, \theta_0; \theta) = -\mathbb{E}_{\boldsymbol{x}, \boldsymbol{y} \in \mathcal{D}_f} \mathcal{L}_{\text{CE}}(g_{\theta'}(\boldsymbol{x}), \boldsymbol{y}; \theta) + \lambda \|\text{vec}(B)\|_1. \tag{6}$$

After untraining is performed, A and B will contain the modifications applied to layer f to remove the knowledge of $\mathcal{D}_f$ while maintaining that of $\mathcal{D}_r$. The original knowledge of the network, though, will still be accessible through W. During the evaluation, W can be made inaccessible by just collapsing the decomposition settled in Eq. 4 back into a single parameter matrix, as follows:

$$W' \leftarrow W_0 + BA, \;\; f(x) = xW' + b. \tag{7}$$

After performing this operation, the resulting unlearned network will also have the same number of parameters as the pre-trained model. Our approach is also visually depicted in Fig. 2.

3.4 Bounded unlearning loss

To realize proper unlearning, $\mathcal{L}$ should be minimized, zeroing the regularization term and increasing the forget loss as much as possible. However, this involves some drawbacks. Firstly, as we perform gradient ascent, the forget loss is not

bounded, like standard loss functions. Further, as the loss approaches negative infinity, we would end up having $\|\mathcal{L}_{\text{CE}}(\cdot)\| \gg \|\mathcal{R}_{\text{ret}}(\cdot)\|$, losing any numerical guarantee that the regularizer will maintain information on the retaining classes.

To overcome these issues, we propose to minimize the following objective, in which we employ the reciprocal of the forget loss, with a positive sign:

$$\mathcal{L}(\mathcal{D}_f, \theta_0; \theta) = \frac{1}{\mathbb{E}_{\boldsymbol{x},\boldsymbol{y}\in\mathcal{D}_f}\mathcal{L}_{\text{CE}}(g_{\theta'}(\boldsymbol{x}), \boldsymbol{y}; \theta)} + \lambda\|\text{vec}(B)\|_1. \tag{8}$$

As it can be seen, the loss defined above can be minimized towards zero, imposing the unlearning loss to be maximized, and the retain regularizer to be minimized. Through the rest of the paper, the unlearning loss above will be referred to as *bounded unlearning loss*.

4 Experimental Evaluation

4.1 Experimental setting

We conduct a set of different experiments to validate the effectiveness of low-rank unlearning, by comparing with baselines and state-of-the-art approaches.

Backbones. While most of the recent unlearning literature has employed small-sized CNNs [9], we argue that it is crucial to test the effectiveness of unlearning methods over modern image classification architectures. This choice increases the effectiveness of the comparisons by reflecting a scenario closer to a future production-like environment and helps to guide the literature toward developing models that are more useful in real-world applications. Following this line, we employ image classification backbones based on Vision Transformers, which have proven their effectiveness on a wide range of tasks [1, 3, 13]. In particular, we employ the original ViT model [17] in its Tiny and Small versions (*i.e.* ViT-T and ViT-S respectively) and the Swin Transformer architecture [30] in its Small configuration (*i.e.* Swin-S). Following concurrent works that have employed low-rank decompositions for fine-tuning language models [23], we apply the low-rank adapters to each linear layer producing the query, key, and value vectors.

Datasets. Following [11], we perform experiments on the CIFAR-10 dataset and on a modified version of CIFAR-100 where images are grouped in 20 super-classes by considering their semantic similarity. We refer to this modified version as CIFAR-20. Both datasets [27] contain 50,000 training and 10,000 validation samples. In all experiments, we follow the standard splits. Additionally, we extend our analysis on the ImageNet-1k dataset [16] which contains images corresponding to 1,000 different classes. For these experiments, we perform unlearning on 10 random classes[3], using the original splits.

Baselines. To test and compare the effectiveness of the proposed strategies, we employ the following baselines: the *original model*, *i.e.* trained from scratch

[3] The classes that we consider are as follows: `kite`, `mud turtle`, `triceratops`, `scorpion`, `peacock`, `goose`, `jellyfish`, `snail`, `flamingo`, `beagle`.

on the corresponding datasets reported in the tables using the standard cross-entropy loss, without performing any unlearning strategy; the *retrained model*, *i.e.* a model trained from scratch on $\mathcal{D}_r$; the *fine-tuned model*, *i.e.* the original model fine-tuned on $\mathcal{D}_r$. Additionally, we implement two other unlearning baselines typically used in previous works [8, 9], namely *random labels* [22] where we fine-tune the model using randomly assigned labels for samples from $\mathcal{D}_f$, and *negative gradient* [20] in which the model is fine-tuned on $\mathcal{D}_f$ using negative gradients (*i.e.* fine-tuned in the direction of gradient ascent). To validate the effectiveness of our model, we also design three model alternatives to measure the contribution of the proposed low-rank unlearning and bounded forget loss.

Metrics. To evaluate class-wise unlearning, we measure the accuracy scores on both retaining and forget sets of the validation split of each dataset (*i.e.* Acc_r and Acc_f respectively). Ideally, the accuracy on the retaining set should be close to that of the original model, while the accuracy on the forget set should be equal to zero, thus getting close accuracy with the retrained model.

Unlearning details. To test the true capabilities of each of the baselines, we opt for maximizing the performance of each of them by running a separate grid search over their loss weights for each backbone and dataset. This is a reasonable choice, as in practice the data holder could run a similar grid search over its own architecture and data before deploying a model in production. Also, we adopt an early stopping procedure that considers the average between the retain accuracy and the opposite of the forget accuracy (*i.e.* $100 - \text{Acc}_f$), so as to evenly balance between the capabilities of forgetting and those of retaining. This is also different from what has been done in recent works [8] in which the early stopping criterion was set exclusively on Acc_r, thus showcasing the forget capabilities of an approach to the detriment of its retaining effectiveness.

In all experiments, we employ Adam [26] as optimizer with a batch size of 256. We use a learning rate equal to 0.0001 for the baselines employing retaining data. In our setting without the retaining set, instead, we use a learning rate of 0.01 and 0.00005 respectively for the models with and without low-rank fine-tuning. In our complete model, we set the λ regularization weight to 0.001 for ViT-T and 0.0025 for ViT-S and Swin-S. The rank of the decomposition r is always set to 8, as it performed favorably in our initial experiments.

4.2 Utility analysis

A machine unlearning method is effective when the unlearned model contains little or no information about the forget data items contained in $\mathcal{D}_r$. In the following, we evaluate the utility of the different baselines and that of the proposed approach, by also conducting ablation experiments. Results are reported in Table 1, over the three considered backbones and on both datasets.

We begin by considering the retrained model in comparison with the original model, which provides an upper bound in terms of accuracy on both the retaining set and the forget set and which, on the other side, needs access to the full training dataset. We then compare with three unlearning approaches which need

Table 1. Class-wise unlearning performance, comparing our solution with baselines with access to retaining data and different ablations. Column $\mathcal{D}_r$ indicates whether the method needs access to the retain set. Final accuracy scores are obtained by performing an unlearning stage for each of the dataset classes and then averaging the results.

			ViT-T		**ViT-S**		**Swin-S**	
		$\mathcal{D}_r$	$\text{Acc}_r \uparrow$	$\text{Acc}_f \downarrow$	$\text{Acc}_r \uparrow$	$\text{Acc}_f \downarrow$	$\text{Acc}_r \uparrow$	$\text{Acc}_f \downarrow$
CIFAR-10	Original model	-	82.0	82.0	84.0	84.0	89.8	89.8
	Retrained model	✓	80.9	0.0	85.4	0.0	88.8	0.0
	Fine-tuned model	✓	80.2	7.9	81.3	3.0	85.0	2.3
	Random labels [22]	✓	83.0	0.0	85.1	0.0	88.9	0.0
	Negative gradient [20]	✓	84.4	0.0	85.8	0.0	88.9	0.0
	Negative gradient w/ L_1 regularization	✗	80.8	0.3	82.2	1.0	85.4	2.1
	Negative gradient w/ low-rank	✗	80.9	**0.1**	82.5	0.9	85.4	1.8
	Bounded loss w/ L_1 regularization	✗	81.2	**0.1**	82.3	**0.8**	85.5	1.4
	Bounded loss w/ low-rank (Ours)	✗	**81.9**	**0.1**	**83.5**	**0.8**	**86.0**	**0.8**
CIFAR-20	Original model	-	67.0	67.0	71.9	71.9	74.4	74.4
	Retrained model	✓	64.2	0.0	69.7	0.0	72.7	0.0
	Fine-tuned model	✓	64.5	8.2	67.2	8.6	68.3	4.6
	Random labels [22]	✓	66.2	0.0	70.8	0.0	73.2	0.0
	Negative gradient [20]	✓	67.6	0.0	71.4	0.0	72.2	0.0
	Negative gradient w/ L_1 regularization	✗	62.9	1.1	68.0	1.2	67.9	3.8
	Negative gradient w/ low-rank	✗	63.0	1.0	67.8	1.0	67.9	3.8
	Bounded loss w/ L_1 regularization	✗	63.1	1.2	67.9	0.8	68.0	3.7
	Bounded loss w/ low-rank (Ours)	✗	**63.5**	**0.9**	**68.2**	**0.8**	**68.2**	**3.4**

access to the retaining data as well, and which therefore operate on a setting that is easier than the one on which we operate. Namely, we compare the model trained with random labels and one trained with negative gradient.

Firstly, we notice that both the random labels approach and the negative gradient approach are effective in forgetting the data contained in $\mathcal{D}_f$, as testified by their zero accuracies on the forget class. We also notice that fine-tuning the original model on $\mathcal{D}_r$ is effective in erasing the information on $\mathcal{D}_f$ to some degree, even though the baseline fails to reach a zero accuracy and also takes more training time, as already noted by previous literature [9]. Also, the random labels approach struggles to maintain good retain accuracy, which can be explained by the significant ground-truth noise caused by the model. The negative gradient approach, instead, is effective at both forgetting data and maintaining the accuracy on other classes and reaches an accuracy on $\mathcal{D}_r$ which is comparable, or even superior in some cases, to that of the retrained model.

We then turn our attention to the "no retain set" scenario, in which the model has no access to $\mathcal{D}_r$, where we investigate the performance of the negative gradient approach, that of a model trained with the bounded unlearning loss, that of a model trained with low-rank unlearning, and that of our complete model. For the negative gradient baseline, we employ the negative gradient loss on $\mathcal{D}_f$, in conjunction with an L_1 regularization on weight change, without employing low-rank matrices. This baseline, while being consistent for comparing with our final model, is also in line with recent works that demonstrated the effectiveness of sparsity in unlearning [25].

Table 2. Single-class unlearning performance on 10 randomly selected classes from ImageNet-1k, using ViT-Small as backbone. Averaged results and standard deviations are reported in the rightmost columns.

		Class 1		**Class 2**		**Class 3**		**Class 4**		**Class 5**	
	$\mathcal{D}_r$	$\text{Acc}_r \uparrow$	$\text{Acc}_f \downarrow$	$\text{Acc}_r \uparrow$	$\text{Acc}_f \downarrow$	$\text{Acc}_r \uparrow$	$\text{Acc}_f \downarrow$	$\text{Acc}_r \uparrow$	$\text{Acc}_f \downarrow$	$\text{Acc}_r \uparrow$	$\text{Acc}_f \downarrow$
Original model	-	86.0	92	86.0	94.0	84.5	74.0	83.5	76.0	85.0	92.0
Random labels [22]	✓	58.9	0.0	62.7	0.0	83.8	0.0	79.3	0.0	79.3	0.0
Negative gradient [20]	✓	77.5	2.0	74.4	3.2	65.3	0.0	70.7	0.0	62.2	0.0
Bounded loss w/ low-rank (Ours)	✗	68.0	0.0	61.3	6.0	70.4	0.0	74.9	0.0	69.8	0.0

		Class 6		**Class 7**		**Class 8**		**Class 9**		**Class 10**		**Avg (ViT-Small)**	
	$\mathcal{D}_r$	$\text{Acc}_r \uparrow$	$\text{Acc}_f \downarrow$	$\text{Acc}_r \uparrow$	$\text{Acc}_f \downarrow$	$\text{Acc}_r \uparrow$	$\text{Acc}_f \downarrow$	$\text{Acc}_r \uparrow$	$\text{Acc}_f \downarrow$	$\text{Acc}_r \uparrow$	$\text{Acc}_f \downarrow$	$\text{Acc}_r \uparrow$	$\text{Acc}_f \downarrow$
Original model	-	84.0	82.0	86.0	74.0	81.9	75.9	81.9	80.0	82.9	93.9	84.2±1.5	83.4±8.2
Random labels [22]	✓	76.0	0.0	78.7	8.0	77.8	0.0	78.0	18.0	61.3	0.0	70.6±8.5	4.6±2.7
Negative gradient [20]	✓	59.5	4.0	69.8	0.0	68.9	0.0	78.6	0.0	48.9	0.0	67.7±8.4	0.9±1.5
Bounded loss w/ low-rank (Ours)	✗	68.7	2.0	75.1	0.0	73.1	0.0	79.1	0.0	70.4	0.0	71.1±4.6	3.0±6.6

Employing an effective unlearning approach such as the negative gradient one, without having access to the retain set, results in a significant lowering of the retain accuracy, of around one accuracy point on all backbones and datasets. The addition of low-rank fine-tuning and the bounded unlearning loss, instead, provide a good recovery of the retaining capabilities of the models, without compromising the forget accuracy, or even enhancing it in some cases. On CIFAR-10 and ViT-T, the combination of bounded loss and low-rank learning enhances the retain accuracy by 1.1 points, while keeping the same forget accuracy, while on ViT-S it increases the retain accuracy by 1.3 points and improves the forget accuracy by 0.2 points. The same applies to the Swin-S model, where low-rank unlearning significantly increases unlearning performance with respect to the negative gradient baseline. The same can be observed on CIFAR-20, and over all the three considered backbones. For instance, low-rank unlearning on ViT-T increases the retain accuracy by 0.6 points, while obtaining a 0.9 forget accuracy.

In Table 2, we instead report the results on ImageNet-1k, considering the ViT-S model and 10 randomly selected classes. In this setting, we compare our model with the negative gradient and random labels baselines, which both leverage the retain set during unlearning. For completeness, we also show the results of the original model which represents an upper bound reference. From the results, it can be noticed that performing unlearning on the ImageNet-1k dataset is in general more challenging: while all considered models can adequately unlearn the selected classes, they experience some performance drops on the retain set. It is worth noting, however, that our model can achieve competitive retain accuracy scores, performing better or on par than the two considered competitors which both have access to the retain set during unlearning. All reported results outline that our method achieves the utility guarantee effectively and low-rank decomposition is a viable solution to perform unlearning without retaining data.

4.3 Visualizations

Embedding space visualizations. To better visualize the effect of unlearning on the decision space of the network, we report t-SNE [32] visualizations of the embedding space produced by the classification layer of the ViT-Tiny and ViT-Small models, both unlearned on CIFAR-10. For comparison purposes, we report

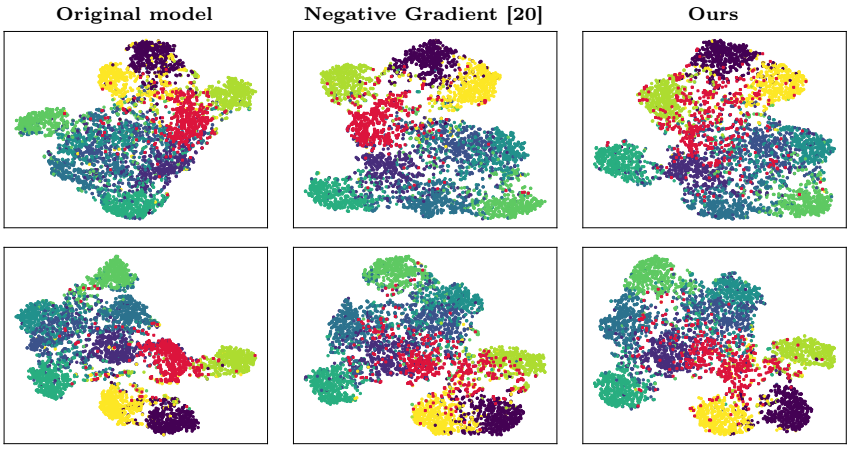

Fig. 3. Visualization of the embedding space of pre-trained models and unlearned models on CIFAR-10 using the ViT-Tiny (top) and ViT-Small (bottom) backbones. Samples from the unlearned class are represented with red markers.

the visualization obtained by the pre-trained model, by the negative gradient approach (which employs retaining data), and by low-rank unlearning. As it can be seen from Fig. 3, low-rank unlearning brings the embedding of unlearned samples toward other classes, thus realizing the unlearning objective. Noticeably, unlearned samples are moved to the embedding space of multiple classes, which is a valuable effect. The opposite, indeed, could represent the Streisand effect and provide more information about the forgetting data [20]. We observe that this can happen in the case of the negative gradient baseline, especially with the ViT-Tiny backbone, despite this baseline having access to retaining data. Low-rank unlearning, instead, appears to be less prone to collapsing unwanted data in the embedding space of a single class.

Further, we can notice that the clusters representing the other classes have remained compact after unlearning, which testifies that their knowledge has been retained. In particular, we observe that there is no significant difference between retained clusters in negative gradient and those in low-rank unlearning. Therefore, low-rank unlearning can effectively unlearn the embedding of a class, while correctly maintaining the knowledge of retained classes.

Attention maps. We also report the attention maps of models untrained with our approach and with other approaches from the literature. In particular, we employ Grad-CAM visualizations [40], which have been originally developed for convolutional neural networks and which can seamlessly be adapted to Vision Transformers [37]. We do this by reducing the stride of the first convolutional layer of a ViT, so as to have an attention map with higher resolution. The maps represent the areas of the input image that the network has paid more attention to when predicting the final output distribution. Results are reported in Fig. 4, where we can observe that the attention maps produced by low-rank unlearning are significantly sparser than those produced by the negative gradient baseline,

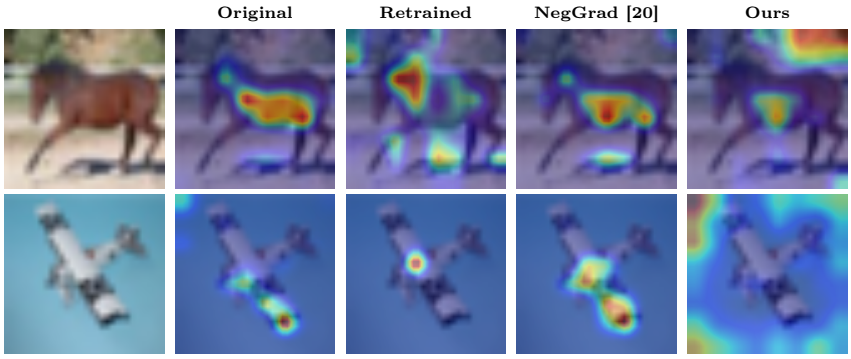

Fig. 4. Grad-CAM [40] attention visualizations of different unlearning methods.

and tend to shift the attention from the foreground object to the background, in a manner that closely resembles the behavior of the retrained model. These results confirm that low-rank unlearning is effective in removing knowledge of the unwanted class, and also that models fine-tuned with our approach closely resemble the models retrained without the unwanted data.

4.4 Computational analysis

One of the most significant benefits of low-rank unlearning is that we greatly reduce memory occupation and storage requirements, and we can also reduce the computation times required to unlearn a given class. In particular, for a Vision Transformer trained with Adam, low-rank unlearning reduces the VRAM requirement by up to 2/3 with $r = 8$. Further, we also observed a reduction in backward times without increasing the number of iterations needed to bring the model to early stopping. The detailed unlearning times are reported in Table 3, in which we measure the number of seconds required to unlearn a single class, averaging the results on all CIFAR-10 classes. We compare the unlearning times of our complete model with those of the baseline without low-rank decomposition and bounded unlearning loss (*i.e.* negative gradient with L_1 regularization), using a single P100 GPU to run the experiments. As it can be seen, our proposal does not negatively impact unlearning times but on the contrary, it contributes to improving the efficiency of model training when employing both ViT-Tiny and ViT-Small model versions, thus further confirming the appropriateness of our solution. It is also worth noting that at test time our model has the same number of parameters as the original model. This guarantees that we do not introduce any additional latency during inference compared to a retrained model.

4.5 Few-shot unlearning analysis

Finally, we analyze the impact of using a reduced number of samples from $\mathcal{D}_f$ to perform unlearning. The results are shown in Fig. 5 using a variable number of samples from the CIFAR-10 forget set. Also in this case, we compare our model

Table 3. Unlearning times measured as the average number of seconds required to unlearn a single class. Results are reported on the CIFAR-10 dataset.

		Unlearning Time (s)	
	$\mathcal{D}_r$	ViT-Tiny	ViT-Small
Negative gradient w/ L_1 regularization	✗	7.96	10.38
Bounded loss w/ low-rank (Ours)	✗	6.83	9.16

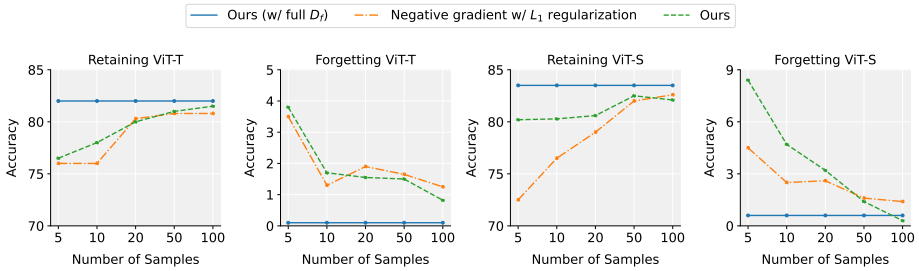

Fig. 5. Retaining and forget accuracy scores when varying the number of forget samples for each class. Results are reported on the CIFAR-10 dataset.

with the negative gradient baseline with L_1 regularization and also report the accuracy upper bounds obtained by our model trained using all samples in D_f. Notably, using 100 forget samples per class (*i.e.* instead of 5,000 as in the full CIFAR-10 forget set) does not significantly deteriorate the performance. Both ViT-Tiny and ViT-Small models achieve better retaining accuracy scores when using our configuration compared to the baseline. In terms of forget accuracy, our model can effectively forget the selected class, especially with 20, 50, and 100 forget samples. When instead using a very limited number of forget samples per class, the accuracy scores are comparable or slightly worse than those obtained by the baseline, which however loses in terms of retaining capabilities.

5 Conclusion

We have presented low-rank unlearning. Our approach removes the knowledge of entire classes from a pre-trained neural network by learning a low-rank adaptation of the network weights, which is then accumulated into the original weights at test time. By leveraging a sparsity regularization, our approach does not need access to the retain dataset, making it suitable for production-like environments. Further, compared to previous approaches, it requires less computational resources, less memory allocation, and fewer storage requirements at training time. Extensive experimental results have demonstrated its performance in unlearning of modern image classification architectures. We envision our work as a step in the direction of efficient and effective unlearning.

Acknowledgments This work has been conducted under a research grant co-funded by Leonardo S.p.A. and supported by the EU Horizon project "ELIAS - European Lighthouse of AI for Sustainability" (No. 101120237).

References

1. Barsellotti, L., Amoroso, R., Cornia, M., Baraldi, L., Cucchiara, R.: Training-Free Open-Vocabulary Segmentation with Offline Diffusion-Augmented Prototype Generation. In: CVPR (2024)
2. Baumhauer, T., Schöttle, P., Zeppelzauer, M.: Machine unlearning: Linear filtration for logit-based classifiers. Machine Learning **111**(9), 3203–3226 (2022)
3. Bontempo, G., Porrello, A., Bolelli, F., Calderara, S., Ficarra, E.: DAS-MIL: Distilling Across Scales for MIL classification of histological WSIs. In: MICCAI (2023)
4. Bourtoule, L., Chandrasekaran, V., Choquette-Choo, C.A., Jia, H., Travers, A., Zhang, B., Lie, D., Papernot, N.: Machine unlearning. In: IEEE S&P (2021)
5. Caffagni, D., Cocchi, F., Barsellotti, L., Moratelli, N., Sarto, S., Baraldi, L., Baraldi, L., Cornia, M., Cucchiara, R.: The Revolution of Multimodal Large Language Models: A Survey. In: ACL Findings (2024)
6. Caffagni, D., Cocchi, F., Moratelli, N., Sarto, S., Cornia, M., Baraldi, L., Cucchiara, R.: Wiki-LLaVA: Hierarchical Retrieval-Augmented Generation for Multimodal LLMs. In: CVPR Workshops (2024)
7. Cao, Y., Yang, J.: Towards Making Systems Forget with Machine Unlearning. In: IEEE S&P (2015)
8. Cha, S., Cho, S., Hwang, D., Lee, H., Moon, T., Lee, M.: Learning to Unlearn: Instance-wise Unlearning for Pre-trained Classifiers. In: AAAI (2024)
9. Chen, M., Gao, W., Liu, G., Peng, K., Wang, C.: Boundary Unlearning. In: CVPR (2023)
10. Chen, M., Zhang, Z., Wang, T., Backes, M., Humbert, M., Zhang, Y.: When machine unlearning jeopardizes privacy. In: ACM CCS (2021)
11. Chundawat, V.S., Tarun, A.K., Mandal, M., Kankanhalli, M.: Can Bad Teaching Induce Forgetting? Unlearning in Deep Networks using an Incompetent Teacher. In: AAAI (2023)
12. Chundawat, V.S., Tarun, A.K., Mandal, M., Kankanhalli, M.: Zero-Shot Machine Unlearning. IEEE Trans. IFS **18**, 2345–2354 (2023)
13. Cornia, M., Baraldi, L., Cucchiara, R.: Explaining Transformer-based Image Captioning Models: An Empirical Analysis. AI Communications **35**(2), 111–129 (2022)
14. Cucchiara, R., Baraldi, L., Cornia, M., Sarto, S.: Video Surveillance and Privacy: A Solvable Paradox? Computer **57**(3), 91–100 (2024)
15. Dang, Q.V.: Right to Be Forgotten in the Age of Machine Learning. In: ICADS (2021)
16. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: A large-scale hierarchical image database. In: CVPR (2009)
17. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., et al.: An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In: ICLR (2021)
18. Goddard, M.: The EU General Data Protection Regulation (GDPR): European Regulation that has a Global Impact. IJMR **59**(6), 703–705 (2017)
19. Golatkar, A., Achille, A., Ravichandran, A., Polito, M., Soatto, S.: Mixed-privacy forgetting in deep networks. In: CVPR (2021)
20. Golatkar, A., Achille, A., Soatto, S.: Eternal sunshine of the spotless net: Selective forgetting in deep networks. In: CVPR (2020)
21. Graves, L., Nagisetty, V., Ganesh, V.: Amnesiac machine learning. In: AAAI (2021)
22. Hayase, T., Yasutomi, S., Katoh, T.: Selective Forgetting of Deep Networks at a Finer Level than Samples. arXiv preprint arXiv:2012.11849 (2020)

23. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: LoRA: Low-Rank Adaptation of Large Language Models. arXiv preprint arXiv:2106.09685 (2021)
24. Izzo, Z., Smart, M.A., Chaudhuri, K., Zou, J.: Approximate data deletion from machine learning models. In: AISTATS (2021)
25. Jia, J., Liu, J., Ram, P., Yao, Y., Liu, G., Liu, Y., Sharma, P., Liu, S.: Model Sparsity Can Simplify Machine Unlearning. In: NeurIPS (2023)
26. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. In: ICLR (2015)
27. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images (2009)
28. Lin, S., Zhang, X., Chen, C., Chen, X., Susilo, W.: ERM-KTP: Knowledge-Level Machine Unlearning via Knowledge Transfer. In: CVPR (2023)
29. Liu, J., Xue, M., Lou, J., Zhang, X., Xiong, L., Qin, Z.: MUter: Machine Unlearning on Adversarially Trained Models. In: ICCV (2023)
30. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin Transformer: Hierarchical vision transformer using shifted windows. In: ICCV (2021)
31. Luo, Z., Xu, X., Liu, F., Koh, Y.S., Wang, D., Zhang, J.: Privacy-preserving low-rank adaptation for latent diffusion models. arXiv preprint arXiv:2402.11989 (2024)
32. Van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. JMLR **9**(11) (2008)
33. Neel, S., Roth, A., Sharifi-Malvajerdi, S.: Descent-to-Delete: Gradient-Based Methods for Machine Unlearning. In: ALT (2021)
34. Nguyen, Q.P., Low, B.K.H., Jaillet, P.: Variational Bayesian Unlearning. In: NeurIPS (2020)
35. Nguyen, T.T., Huynh, T.T., Nguyen, P.L., Liew, A.W.C., Yin, H., Nguyen, Q.V.H.: A Survey of Machine Unlearning. arXiv preprint arXiv:2209.02299 (2022)
36. Pawelczyk, M., Neel, S., Lakkaraju, H.: In-Context Unlearning: Language Models as Few Shot Unlearners. arXiv preprint arXiv:2310.07579 (2023)
37. Poppi, S., Cornia, M., Baraldi, L., Cucchiara, R.: Revisiting the Evaluation of Class Activation Mapping for Explainability: A Novel Metric and Experimental Analysis. In: CVPR Workshops (2021)
38. Poppi, S., Poppi, T., Cocchi, F., Cornia, M., Baraldi, L., Cucchiara, R.: Safe-CLIP: Removing NSFW Concepts from Vision-and-Language Models. In: ECCV (2024)
39. Poppi, S., Sarto, S., Cornia, M., Baraldi, L., Cucchiara, R.: Multi-Class Unlearning for Image Classification via Weight Filtering. IEEE Intelligent Systems (2024)
40. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-CAM: Visual Explanations From Deep Networks via Gradient-Based Localization. In: ICCV (2017)
41. Sun, Y., Li, Z., Li, Y., Ding, B.: Improving loRA in privacy-preserving federated learning. In: ICLR (2024)
42. Tarun, A.K., Chundawat, V.S., Mandal, M., Kankanhalli, M.: Fast Yet Effective Machine Unlearning. IEEE Trans. NNLS (2023)
43. de la Torre, L.: A Guide to the California Consumer Privacy Act of 2018. Available at SSRN 3275571 (2018)
44. Wu, Y., Dobriban, E., Davidson, S.: Deltagrad: Rapid retraining of machine learning models. In: ICML (2020)
45. Yoon, Y., Nam, J., Yun, H., Kim, D., Ok, J.: Few-Shot Unlearning by Model Inversion. arXiv preprint arXiv:2205.15567 (2022)