Model Sparsity Can Simplify Machine Unlearning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Recent data regulations necessitate machine unlearning (MU): The removal of the effect of specific examples from the model. While *exact* unlearning is possible by conducting a model retraining with the remaining data from scratch, its computational cost has led to the development of approximate but efficient unlearning schemes. Beyond data-centric MU solutions, we advance MU through a novel model-based viewpoint: sparsification via weight pruning. Our results in both theory and practice indicate that model sparsity can boost the multi-criteria unlearning performance of an approximate unlearner, closing the approximation gap, while continuing to be efficient. With this insight, we develop two new sparsity-aware unlearning meta-schemes, termed 'prune first, then unlearn' and 'sparsity-aware unlearning'. Extensive experiments show that our findings and proposals consistently benefit MU in various scenarios including class-wise data scrubbing, random data scrubbing, and backdoor data forgetting. One highlight is the 77% unlearning efficacy gain of fine-tuning (one of the simplest approximate unlearning methods) in the proposed sparsity-aware unlearning paradigm.

1 Introduction

Machine unlearning (**MU**) initiates a reverse learning process to scrub the influence of data points from a trained machine learning (**ML**) model. It was introduced to avoid information leakage about private data upon completion of training [1–3], particularly in compliance with legislation like 'the right to be forgotten' [4] in General Data Protection Regulation (GDPR) [5]. The *direct but optimal* unlearning approach is *exact unlearning* to *retrain* ML models from scratch using the training set after removing data points to be scrubbed. Although retraining yields the *ground-truth* unlearning strategy, it is most computationally intensive. Thus, the development of *approximate but fast* unlearning methods has become a major focus in research [6–10].

Despite the computational benefits of approximate unlearning, it often lacks a strong guarantee on the effectiveness of unlearning, resulting in a performance gap with exact unlearning [11]. In particular, we encounter two main challenges. *First*, the performance of approximate unlearning can heavily rely on the configuration of algorithmic parameters. For example, the Fisher forgetting method [12] needs to carefully tune the Fisher information regularization parameter in each data-model setup. Second, the effectiveness of an approximate scheme can vary significantly across the multiple unlearning evaluation criteria and their tradeoffs are not well understood. For example, high 'efficacy' (ability to protect the privacy of the scrubbed data) *neither* implies nor precludes high 'fidelity' (accuracy on the remaining dataset) [9]. This raises our **driving question (Q)** below:

> ***(Q)*** *Is there a theoretically-grounded and broadly-applicable method to improve approximate unlearning across different unlearning criteria?*

To address **(Q)**, we advance MU through a fresh and novel viewpoint: *model sparsification*. Our key finding is that model sparsity (achieved by weight pruning) can significantly reduce the gap between approximate unlearning and exact unlearning; see **Fig. 1** for the schematic overview of our proposal and highlighted experiment results.

Model sparsification (or weight pruning) has been extensively studied in the literature [13–19], focusing on the interrelation between model compression and generalization. For example, the notable lottery ticket hypothesis (**LTH**) [15] demonstrated the existence of a sparse subnetwork (the so-called 'winning ticket') that matches or even exceeds the test accuracy of the original dense model. In addition to generalization, the impact of pruning has also been investigated on model robustness [20–22], fairness [23, 24], interpretability [25, 26], loss landscape [16, 26], and privacy [27, 28]. In particular, the privacy gains from pruning [27, 28] imply connections between data influence and model sparsification.

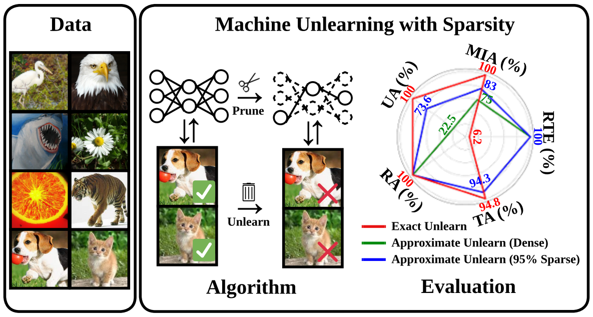

Figure 1: Schematic overview of our proposal on model sparsity-driven MU. Evaluation at-a-glance shows the performance of three unlearning methods (retraining-based exact unlearning, finetuning-based approximate unlearning [12], and proposed finetuning-based unlearning on 95%-sparse model) under five metrics, unlearning accuracy (UA), membership inference attack (MIA)-based unlearning efficacy, accuracy on remaining data (RA), testing accuracy (TA), and run-time efficiency (RTE); see summary in **Tab. 1**. The unlearning scenario is given by class-wise forgetting, where data points of a single class are scrubbed. Each performance metric is normalized to $[0, 1]$ by dividing the best result (100%) across unlearning methods. Results indicate that model sparsity reduces the gap between exact and approximate MU.

More recently, a few works [29, 30] attempted to draw insights from pruning for unlearning. In [29], removing channels of a deep neural network (**DNN**) showed an unlearning benefit in federated learning. And in [30], filter pruning was introduced in lifelong learning to detect 'pruning identified exemplars' [31] that are easy to forget. However, different from the above literature that customized model pruning for a specific unlearning application, our work systematically and comprehensively explores and exploits the foundational connections between unlearning and pruning. We summarize our **contributions** below.

• First, we provide a holistic understanding of MU across the full training/evaluation stack.

• Second, we draw a tight connection between MU and model pruning and show, in both theory and practice, that model sparsity (achieved by pruning) helps us to close the gap between approximate unlearning and exact unlearning.

• Third, we develop a new MU paradigm termed 'prune first, then unlearn', and investigate the influence of pruning methods in the performance of unlearning. As an extension, we also develop a new 'sparsity-aware unlearning' framework, which integrates pruning into unlearning, without requiring model sparsification before unlearning.

• Lastly, we conduct extensive experiments across different datasets, models, and unlearning scenarios. We consistently find that model sparsity is the key to improving MU, as exemplified in **Fig. 1**.

2 Revisiting MU Training and Evaluation

Problem setup of MU. MU aims to remove (or scrub) the influence of some targeted training data on a trained ML model [1, 2]. Let $\mathcal{D} = \{\mathbf{z}_i\}_{i=1}^N$ be a (training) dataset of N data points, with label information encoded for supervised learning. $\mathcal{D}_\mathrm{f} \subseteq \mathcal{D}$ represents a subset whose influence we want to scrub, termed the **forgetting dataset**. Accordingly, the complement of $\mathcal{D}_\mathrm{f}$ is the **remaining dataset**, *i.e.*, $\mathcal{D}_\mathrm{r} = \mathcal{D} \setminus \mathcal{D}_\mathrm{f}$. We denote by $\boldsymbol{\theta}$ the model parameters, and $\boldsymbol{\theta}_\mathrm{o}$ the **original model** trained on the entire training set $\mathcal{D}$ using *e.g.*, empirical risk minimization (**ERM**). Similarly, we denote by $\boldsymbol{\theta}_\mathrm{u}$ an **unlearned model**, obtained by a scrubbing algorithm, after removing the influence of $\mathcal{D}_\mathrm{f}$ from the trained model $\boldsymbol{\theta}_\mathrm{o}$. The **problem of MU** is to find an accurate and efficient scrubbing mechanism to generate $\boldsymbol{\theta}_\mathrm{u}$ from $\boldsymbol{\theta}_\mathrm{o}$.

In the literature [2, 7, 12], the choice of the forgetting dataset $\mathcal{D}_\mathrm{f}$ specifies different unlearning scenarios. There exist three main categories. First, *class-wise forgetting* [7, 12] refers to unlearning

$\mathcal{D}_\mathrm{f}$ consisting of training data points of an entire class. Second, *random data forgetting (per class)* [12] refers to unlearning $\mathcal{D}_\mathrm{f}$ given by a subset of data points randomly selected from a single class. Third, *random data forgetting (all classes)* corresponds to unlearning $\mathcal{D}_\mathrm{f}$ given by a subset of random data points drawn from all classes.

Exact and approximate MU methods. The *exact unlearning* method refers to *retraining* the model parameters from *scratch* over the remaining dataset $\mathcal{D}_\mathrm{r}$. Although retraining from scratch (that we term **Retrain**) is optimal for MU, it entails a large computational overhead, particularly for DNN training. This problem is alleviated by *approximate unlearning*, an easy-to-compute proxy for Retrain, which has received growing attention. Yet, the boosted computation efficiency comes at the cost of MU's efficacy. We next review some commonly-used approximate unlearning methods that we improve in the sequel by leveraging sparsity.

✦ *Fine-tuning (**FT**)* [6, 12]: Different from Retrain, FT fine-tunes the pre-trained model $\boldsymbol{\theta}_\mathrm{o}$ on $\mathcal{D}_\mathrm{r}$ using a few training epochs to obtain $\boldsymbol{\theta}_\mathrm{u}$. The rationale behind FT is that fine-tuning on $\mathcal{D}_\mathrm{r}$ initiates the catastrophic forgetting of the model over $\mathcal{D}_\mathrm{f}$ like continual learning [32].

✦ *Gradient ascent (**GA**)* [7, 8]: GA reverses the model training on $\mathcal{D}_\mathrm{f}$ by adding the corresponding gradients back to $\boldsymbol{\theta}_\mathrm{o}$, *i.e.*, moving $\boldsymbol{\theta}_\mathrm{o}$ in the direction of increasing loss for data points to be scrubbed.

✦ *Fisher forgetting (**FF**)* [9, 12]: FF adopts an additive Gaussian noise to 'perturb' $\boldsymbol{\theta}_\mathrm{o}$ towards exact unlearning. Here the Gaussian distribution has zero mean and covariance determined by the 4th root of Fisher Information matrix with respect to (w.r.t.) $\boldsymbol{\theta}_\mathrm{o}$ on $\mathcal{D}_\mathrm{r}$. We note that the computation of the 4th root of Fisher Information matrix could lower the efficiency of FF on GPUs; see [12] for details.

✦ *Influence unlearning (**IU**)* [10, 33]: IU leverages the influence function approach [34] to characterize the change of $\boldsymbol{\theta}_\mathrm{o}$ if a training point is removed from the training loss. IU estimates the change in model parameters from $\boldsymbol{\theta}_\mathrm{o}$ to $\boldsymbol{\theta}_\mathrm{u}$, *i.e.*, $\boldsymbol{\theta}_\mathrm{u} - \boldsymbol{\theta}_\mathrm{o}$, as derived in Proposition 1.

Proposition 1 *Given the weighted ERM training, $\boldsymbol{\theta}(\mathbf{w}) = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N}[w_i \ell_i(\boldsymbol{\theta}, \mathbf{z}_i)]$ where $w_i \in [0,1]$ is the influence weight associated with the data point $\mathbf{z}_i$ and $\mathbf{1}^T\mathbf{w} = 1$, the model update from $\boldsymbol{\theta}_\mathrm{o}$ to $\boldsymbol{\theta}(\mathbf{w})$ yields*

$$\Delta(\mathbf{w}) := \boldsymbol{\theta}(\mathbf{w}) - \boldsymbol{\theta}_\mathrm{o} \approx \mathbf{H}^{-1}\nabla_{\boldsymbol{\theta}} L(\mathbf{1}/N - \mathbf{w}, \boldsymbol{\theta}_\mathrm{o}), \tag{1}$$

where $\mathbf{H}^{-1}$ is the inverse of the Hessian $\nabla_{\boldsymbol{\theta},\boldsymbol{\theta}} L(\mathbf{1}/N, \boldsymbol{\theta}_\mathrm{o})$ evaluated at $\boldsymbol{\theta}_\mathrm{o}$, and $\nabla_{\boldsymbol{\theta}} L$ is the gradient of L. When scrubbing $\mathcal{D}_\mathrm{f}$, the unlearned model is given by $\boldsymbol{\theta}_\mathrm{u} = \boldsymbol{\theta}_\mathrm{o} + \Delta(\mathbf{w}_\mathrm{MU})$, where $\mathbf{w}_\mathrm{MU} = \mathbf{1}_{\mathcal{D}_\mathrm{r}}/\mathrm{card}(\mathcal{D}_\mathrm{r})$, and $\mathcal{D}_\mathrm{r} = \mathcal{D} \setminus \mathcal{D}_\mathrm{f}$.

Proof: We derive (1) using an implicit gradient approach; See Appendix 1. □

It is also worth noting that we have taken into consideration the weight normalization effect $\mathbf{1}^T\mathbf{w} = 1$ in (1). This is different from existing work like [10, Sec. 3] using Boolean or unbounded weights. In practice, we found that IU with weight normalization can improve the unlearning performance. Furthermore, to update the model influence given by (1), one needs to acquire the second-order information in the form of inverse-Hessian gradient product. Yet, the exact computation is prohibitively expensive. To overcome this issue, we use the first-order WoodFisher approximation [35] to estimate the inverse-Hessian gradient product.

Towards a 'full-stack' MU evaluation. Existing work has assessed MU performance from different aspects [7, 8, 12]. Yet, a single performance metric may provide a limited view of MU [11]. By carefully reviewing the prior art, we perform a multi-faceted assessment based on the following empirical metrics.

✦ *Unlearning accuracy (**UA**)*: We define $\mathrm{UA}(\boldsymbol{\theta}_\mathrm{u}) = 1 - \mathrm{Acc}_{\mathcal{D}_\mathrm{f}}(\boldsymbol{\theta}_\mathrm{u})$ to characterize the *efficacy* of MU in the accuracy dimension, where $\mathrm{Acc}_{\mathcal{D}_\mathrm{f}}(\boldsymbol{\theta}_\mathrm{u})$ is the accuracy of $\boldsymbol{\theta}_\mathrm{u}$ on the forgetting dataset $\mathcal{D}_\mathrm{f}$ [7, 12]. Note that a better UA of an approximate unlearning method corresponds to a smaller gap with UA of the gold-standard retrained model (Retrain). This also applies to other metrics.

Table 1: Summary of approximate unlearning methods considered in this work. The marker '✓' denotes the metric used in the corresponding references. The number in RTE demonstrates the run-time cost reduction over the computation time of Retrain, based on our empirical studies in Sec. 5 on (CIFAR-10, ResNet-18). As will be evident later, although GA seems better than ours in terms of RTE, it is less effective in unlearning.

Unlearning	Evaluation metrics					Representative work
Methods	UA	MIA-Efficacy	RA	TA	RTE	
FT	✓		✓	✓	0.06×	[6, 12]
GA	✓	✓	✓	✓	0.02×	[7, 8]
FF	✓		✓	✓	0.9 ×	[9, 12]
IU	✓			✓	0.08×	[10, 33]
Ours	✓	✓	✓	✓	0.07×	This work

✦ *Membership inference attack (MIA) on $\mathcal{D}_\mathrm{f}$ (**MIA-Efficacy**)*: This is another metric to assess the *efficacy* of unlearning, achieved by MIA to determine whether or not a data point in $\mathcal{D}_\mathrm{f}$ belongs to the training set of $\boldsymbol{\theta}_\mathrm{u}$ [36, 37]. A *higher* MIA-Efficacy implies that less information about $\mathcal{D}_\mathrm{f}$ contained in $\boldsymbol{\theta}_\mathrm{u}$.

✦ *Remaining accuracy (**RA**)*: This refers to the accuracy of $\boldsymbol{\theta}_\mathrm{u}$ on $\mathcal{D}_\mathrm{r}$, which reflects the *fidelity* of MU [9] and should be preserved from $\boldsymbol{\theta}_\mathrm{o}$ to $\boldsymbol{\theta}_\mathrm{u}$.

✦ *Testing accuracy (**TA**)*: This measures the *generalization* ability of $\boldsymbol{\theta}_\mathrm{u}$, which is expected at testing time. TA is evaluated on the whole test dataset, except for class-wise forgetting, in which testing data points belonging to the forgetting class are not in the testing scope.

✦ *Run-time efficiency (**RTE**)*: This measures the computation efficiency of an MU method. For example, if we regard the run-time cost of Retrain as the baseline, the computation acceleration gained by different approximate unlearning methods is summarized in Tab. 1.

3 Why Model Sparsity for MU?

This section shows that model sparsity is a missing factor influencing MU.

Model sparsification via weight pruning. It is known that model sparsification could not only facilitate a model's training, inference, and deployment but also benefit model's performance, in particular in generalization. For example, LTH (lottery ticket hypothesis) [15] stated that a trainable sparse sub-model could be identified from the original dense model, with test accuracy on par or even better than the original model.

Fig. 2 shows an example of the pruned model's generalization vs. its sparsity ratio. Here one-shot magnitude pruning (**OMP**) [17] is adopted to obtain sparse models. OMP is the computationally-lightest pruning method, which directly prunes the model weights to the target sparsity ratio based on their magnitudes. As we can see, there exists a graceful sparse regime with lossless testing accuracy.

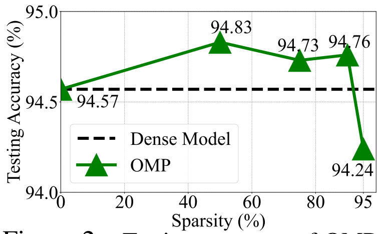

Figure 2: Testing accuracy of OMP-based sparse ResNet-18 and the dense model on CIFAR-10 vs. sparsity.

Gains of MU from sparsity. We first analyze the impact of model sparsity on MU through a lens of *unrolling stochastic gradient descent* (**SGD**) [8], used to derive the *unlearning error* (given by the weight difference between the approximately unlearned model and the gold-standard retrained model) when scrubbing a single data point. However, different from [8], we will infuse the model sparsity into SGD unrolling.

Let us assume a binary mask $\mathbf{m}$ associated with the model parameters $\boldsymbol{\theta}$, where $m_i = 0$ signifies that the ith parameter θ_i is pruned to zero and $m_i = 1$ represents the unmasked θ_i. This sparse pattern $\mathbf{m}$ could be obtained by a weight pruning method, like OMP. Given $\mathbf{m}$, the **sparse model** is $\mathbf{m} \odot \boldsymbol{\theta}$, where $\odot$ denotes the element-wise multiplication. Thudi *et al.* [8] showed that if GA is adopted to scrub a single data point for the original (dense) model $\boldsymbol{\theta}$ (*i.e.*, $\mathbf{m} = \mathbf{1}$), then the gap between GA and Retrain can be approximately bounded in the weight space. **Prop. 2** extends the unlearning error analysis in [8] to a sparse model.

Proposition 2 *Given the model sparse pattern $\mathbf{m}$ and the SGD-based training, the unlearning error of GA (denoted by $e(\mathbf{m})$) can be characterized by the weight distance between the GA-unlearned model and the gold-standard retrained model, yielding the error bound*

$$e(\mathbf{m}) = \mathcal{O}(\eta^2 t \|\mathbf{m} \odot (\boldsymbol{\theta}_t - \boldsymbol{\theta}_0)\|_2 \sigma(\mathbf{m})) \tag{2}$$

where $\mathcal{O}$ is the big O notation that omits the constant factors, η is the learning rate, t is the number of training iterations, $(\boldsymbol{\theta}_t - \boldsymbol{\theta}_0)$ denotes the weight difference at iteration t from its initialization $\boldsymbol{\theta}_0$, $\sigma(\mathbf{m})$ is the largest singular value (σ) of the Hessian $\nabla^2_{\boldsymbol{\theta},\boldsymbol{\theta}} \ell$ (for a training loss ℓ) among the unmasked parameter dimensions, i.e., $\sigma(\mathbf{m}) := \max_j \{\sigma_j(\nabla^2_{\boldsymbol{\theta},\boldsymbol{\theta}} \ell), \text{if } m_j \neq 0\}$.

Proof: See Appendix 2. □

Some key insights can be drawn from **Prop. 2**. *First*, it is clear from (2) that the unlearning error reduces as the model sparsity in $\mathbf{m}$ increases. By contrast, the unlearning error derived in [8] for a dense model (*i.e.*, $\mathbf{m} = \mathbf{1}$) is proportional to the dense model distance $\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_0\|_2$. Thus, model sparsity is beneficial to reducing the gap between (GA-based) approximate and exact unlearning. *Second*, the error bound (2) enables us to relate MU to the spectrum of the Hessian of the loss landscape. The number of active singular values (corresponding to nonzero dimensions in $\mathbf{m}$) decreases when the sparsity grows.

Inspired by Prop. 2, we ask: *Does the above benefit of model sparsification in MU apply to other approximate unlearning methods besides GA?* To tackle this question, we investigate the performance of approximate unlearning on sparse models through our multi-criteria evaluation in **Tab. 1**. **Fig. 3** shows the unlearning efficacy (UA and MIA-Efficacy), fidelity (RA), and generalization (TA) of using approximate unlearning methods to remove the influence of all data points in one class from models pruned to different sparsity levels using OMP. The performance of exact unlearning via Retrain is provided for comparison. As we can see, the efficacy of approximate unlearning is significantly improved as the model sparsity increases, *e.g.*, UA and MIA-Efficacy of using FT over 90% sparsity. By contrast, FT over the dense model (0% sparsity) is the least effective for MU. We also note that the efficacy of Retrain is resilient to model sparsity. Thus, the efficacy gap between exact unlearning and approximate unlearning reduces on sparse models. Further, through the fidelity and generalization lenses, FT and FF yield the RA and TA performance closest to Retrain, compared to other unlearning methods. In the regime of ultra-high sparsity (99%), the efficacy of unlearning yields a tradeoff with RA and TA to some extent.

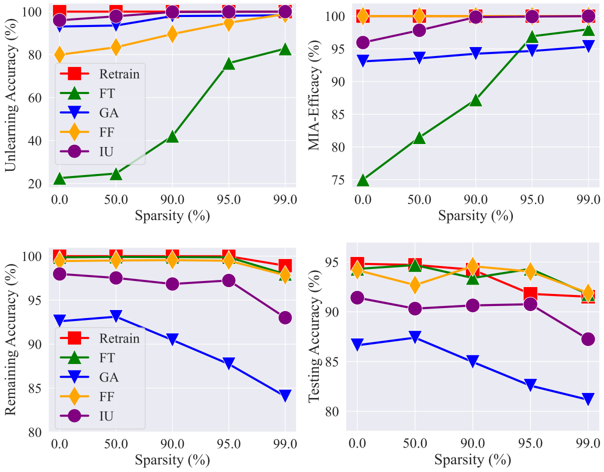

Figure 3: Performance of approximate unlearning (FT, GA, FF, IU) and exact unlearning (Retrain) in efficacy (UA and MIA-Efficacy), fidelity (RA), and generalization (TA) vs. model sparsity (achieved by OMP) in the data-model setup (CIFAR-10, ResNet-18). The unlearning scenario is class-wise forgetting, and the average unlearning performance over 10 classes is reported.

4 Sparsity-Aided Machine Unlearning

We showed that model sparsification could be a principled approach to reduce the gap between approximate unlearning and exact unlearning. This promising finding, however, raises some new questions. First, it remains elusive how the choice of a weight pruning method impacts the unlearning performance. Second, it leaves room for developing sparsity-promoting MU methods that can directly scrub data influence from a dense model.

Prune first, then unlearn: Choice of pruning methods. Our study in Sec. 3 suggested the new MU paradigm ‘prune first, then unlearn’, which shed light on that (approximate) unlearning on a sparse model yields a smaller unlearning error (Prop. 2) and improves the efficacy of MU (Fig. 3). Yet, there exist many ways to find the desired sparse model in addition to OMP. Examples include pruning at random initialization before training [38, 39] and pruning-training simultaneous iterative magnitude pruning (**IMP**) [15]. Thus, the problem of pruning method selection arises for MU. From the viewpoint of MU, the trainer would prioritize a pruning method that satisfies the following criteria: ❶ *least dependence* on the forgetting dataset ($\mathcal{D}_\mathrm{f}$), ❷ *lossless generalization* when pruning, and ❸ *pruning efficiency*. The rationale behind ❶ is that it is desirable *not* to incorporate information of $\mathcal{D}_\mathrm{f}$ when seeking a sparse model for unlearning $\mathcal{D}_\mathrm{f}$. And the criteria ❷ and ❸ ensure that sparsity cannot hamper TA (testing accuracy) and RTE (run-time efficiency). Based on ❶-❸, we propose to use two pruning methods.

✦ **SynFlow** (synaptic flow pruning) [38]: SynFlow provides a (training-free) pruning method at initialization, even without accessing the dataset. Thus, it is uniquely suited for MU to meet the criterion ❶. And SynFlow is easy to compute and yields a generalization improvement over many other pruning-at-initialization methods; see justifications in [38].

✦ **OMP** (one-shot magnitude pruning) [17]: Different from SynFlow, OMP, which we focused on in Sec. 3, is performed over the original model ($\boldsymbol{\theta}_{\mathrm{o}}$). It may depend on the forgetting dataset ($\mathcal{D}_{\mathrm{f}}$), but has a much weaker dependence compared to IMP-based methods. Moreover, OMP is computationally lightest (*i.e.* best for ❸) and yields better generalization than SynFlow [18].

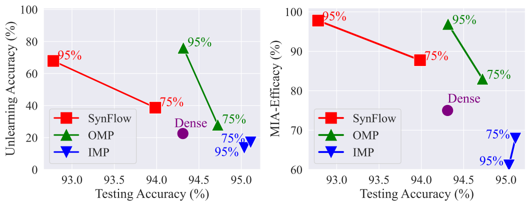

Figure 4: Influence of different pruning methods (SynFlow, OMP, and IMP) in unlearning efficacy (UA and MIA-Efficacy) and generalization (TA) on (CIFAR-10, ResNet-18). **Left**: UA vs. TA. **Right**: MIA-Efficacy vs. TA. Each point is an unlearned dense or sparse model (with 75%, 95% sparsity) using FT. If a pruning method is proper for MU, then its integration with FT should yield unlearned models with higher UA, MIA-Efficacy, and TA.

We also remark that in the research on model pruning, IMP is the predominant approach to find the most accurate sparse models (*i.e.*, best for ❷). However, IMP has the largest computation overhead and the strongest correlation with the training dataset (including $\mathcal{D}_{\mathrm{f}}$). In this sense, IMP may not be a proper pruning choice for MU through the lenses of ❶ and ❸. In **Fig. 4**, we show the efficacy of FT-based unlearning on sparse models generated using different pruning methods (SynFlow, OMP, and IMP). As we can see, unlearning on SynFlow or OMP-generated sparse models yields improved UA and MIA-Efficacy over that on the original dense model and IMP-generated sparse models. This unlearning improvement over the dense model is consistent with Fig. 3. More interestingly, we find that IMP *cannot* benefit the unlearning efficacy, although it leads to the best TA. This is because IMP heavily relies on the training set including forgetting data points, *i.e.*, violating the criterion ❶. We will show a fix to IMP for MU later. Further, when peering into the performance of SynFlow and OMP, we find that the latter typically outperforms the former in TA with similar unlearning efficacy. Thus, unless specified otherwise, OMP is the pruning method we will use in experiments.

Sparsity-aware unlearning on dense models. In the 'prune first, then unlearn' paradigm, model sparsity is given *a priori* for MU. We next study if pruning and unlearning can be carried out simultaneously, without requiring prior knowledge of model sparsity.

Let $L_{\mathrm{u}}(\boldsymbol{\theta}; \boldsymbol{\theta}_{\mathrm{o}}, \mathcal{D}, \mathcal{D}_{\mathrm{f}})$ denote the unlearning objective function of model parameters $\boldsymbol{\theta}$, given the pre-trained state $\boldsymbol{\theta}_{\mathrm{o}}$, the overall training dataset $\mathcal{D}$, and the forgetting dataset $\mathcal{D}_{\mathrm{f}}$. Inspired by optimization with sparsity-inducing penalties [40], we propose integrating an ℓ_1 norm-based sparse penalty into L_{u}. This leads to MU with ℓ_1-norm sparse regularization that we call 'ℓ_1**-sparse MU**':

$$\boldsymbol{\theta}_{\mathrm{u}} = \arg\min_{\boldsymbol{\theta}} L_{\mathrm{u}}(\boldsymbol{\theta}; \boldsymbol{\theta}_{\mathrm{o}}, \mathcal{D}, \mathcal{D}_{\mathrm{f}}) + \gamma \|\boldsymbol{\theta}\|_1, \tag{3}$$

where $\gamma > 0$ is a regularization parameter that controls the sparsity of the model parameters in $\boldsymbol{\theta}$. In (3), if we specify L_{u} as the fine-tuning objective, we then obtain the FT-oriented ℓ_1-sparse MU.

In addition to (3), we can integrate pruning with unlearning via alternating optimization (**AO**) taking inspiration from the IMP algorithm [15]. Yet, different from IMP, we replace the optimization step of retraining non-zero weights with unlearning on non-zero model weights. We term the resulting method '**AO-sparse MU**' and summarize its pipeline (S1)-(S3) below. **(S1)** *Initialize* model $\boldsymbol{\theta} = \boldsymbol{\theta}_{\mathrm{o}}$ and a pruning mask $\mathbf{m} = \mathbf{1}$. **(S2)** *Prune* $p\%$ non-zero parameters in $\boldsymbol{\theta}$ per magnitude. Then, update $\mathbf{m}$ to a sparser mask. **(S3)** *Unlearn* $\mathcal{D}_{\mathrm{f}}$ on the sparse model $\mathbf{m} \odot \boldsymbol{\theta}$ using an approximate unlearning method (*e.g.*, FT) to update the non-zero model parameters and go back to (S2). The above procedure **(S1)**→**(S2)**⇄**(S3)** repeatedly prunes and unlearns the model over multiple rounds (assuming k rounds). Similar to IMP in [15], we adopt a progressive learning process, where each round prunes $p^{1/k}\%$ of the weights on top of the previous round ($p = 20\%$ by default). However, different from IMP, AO-sparse MU is much lighter in computation since approximate unlearning in (S2) takes fewer computation overheads than retraining non-zero model weights in IMP.

5 Experiments

5.1 Experiment setups

Datasets and models. Unless specified otherwise, our experiments will focus on image classification under CIFAR-10 [41] using ResNet-18 [42]. Yet, experiments on additional datasets (CIFAR-100

Table 2: Performance overview of various machine unlearning methods on dense and 95%-sparse models considering different unlearning scenarios: class-wise forgetting, random data forgetting (per class), and random data forgetting (all classes). The forgetting data ratio is 9% of the whole training dataset, the sparse models are obtained using OMP [17], and the unlearning methods and evaluation metrics are summarized in Table 1. Both class-wise forgetting and random data forgetting (per class) are conducted class-wise. The performance is reported in the form $a_{\pm b}$, with mean a and standard deviation b computed over 10 independent trials. A performance gap against Retrain is provided in (●). Note that the better performance of approximate unlearning corresponds to the smaller performance gap with the gold-standard retrained model.

MU	UA		MIA-Efficacy		RA		TA		RTE
	DENSE	**95% Sparsity**	DENSE	**95% Sparsity**	DENSE	**95% Sparsity**	DENSE	**95% Sparsity**	(min)
Class-wise forgetting									
Retrain	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$99.99_{\pm 0.01}$	$94.83_{\pm 0.11}$	$91.80_{\pm 0.89}$	43.23
FT	$22.53_{\pm 8.16}$ (77.47)	$73.64_{\pm 9.46}$ (26.36)	$75.00_{\pm 14.68}$ (25.00)	$83.02_{\pm 16.33}$ (16.98)	$99.87_{\pm 0.04}$ (0.13)	$99.87_{\pm 0.05}$ (0.12)	$94.31_{\pm 0.19}$ (0.52)	$94.32_{\pm 0.12}$ (2.52)	2.52
GA	$93.08_{\pm 2.29}$ (6.92)	$98.09_{\pm 1.11}$ (1.91)	$94.03_{\pm 3.27}$ (5.97)	$97.74_{\pm 2.24}$ (2.26)	$92.60_{\pm 0.25}$ (7.40)	$87.74_{\pm 0.27}$ (12.25)	$86.64_{\pm 0.28}$ (8.19)	$82.58_{\pm 0.27}$ (9.22)	0.33
FF	$79.93_{\pm 8.92}$ (20.07)	$94.83_{\pm 4.29}$ (5.17)	$100.00_{\pm 0.00}$ (0.00)	$100.00_{\pm 0.00}$ (0.00)	$99.45_{\pm 0.24}$ (0.55)	$99.48_{\pm 0.33}$ (0.51)	$94.18_{\pm 0.08}$ (0.65)	$94.04_{\pm 0.10}$ (0.28)	38.91
IU	$87.82_{\pm 2.15}$ (12.18)	$99.47_{\pm 0.15}$ (0.53)	$95.96_{\pm 0.21}$ (4.04)	$99.93_{\pm 0.04}$ (0.07)	$97.98_{\pm 0.21}$ (2.02)	$97.24_{\pm 0.13}$ (2.75)	$91.42_{\pm 0.21}$ (3.41)	$90.76_{\pm 0.18}$ (1.04)	3.25
Random data forgetting (per class)									
Retrain	$56.27_{\pm 0.07}$	$57.86_{\pm 0.05}$	$75.23_{\pm 0.14}$	$76.14_{\pm 0.11}$	$100.00_{\pm 0.00}$	$99.99_{\pm 0.01}$	$89.54_{\pm 0.11}$	$88.41_{\pm 0.89}$	41.63
FT	$1.89_{\pm 0.79}$(54.38)	$19.34_{\pm 1.41}$ (38.52)	$17.11_{\pm 2.21}$ (58.12)	$35.18_{\pm 2.12}$ (40.96)	$99.92_{\pm 0.03}$ (0.08)	$99.21_{\pm 0.05}$ (0.78)	$93.50_{\pm 0.52}$ (3.96)	$91.20_{\pm 0.12}$ (2.79)	2.36
GA	$50.75_{\pm 3.29}$ (5.52)	$59.12_{\pm 3.17}$ (1.26)	$69.27_{\pm 4.27}$ (5.96)	$74.06_{\pm 3.15}$ (2.08)	$98.43_{\pm 0.20}$ (1.57)	$98.59_{\pm 0.17}$ (1.40)	$87.56_{\pm 0.24}$ (1.98)	$87.12_{\pm 0.31}$ (1.29)	0.29
FF	$4.85_{\pm 4.20}$ (51.42)	$6.92_{\pm 3.72}$ (50.94)	$11.29_{\pm 5.12}$ (63.94)	$12.37_{\pm 4.54}$ (63.77)	$97.30_{\pm 0.52}$ (2.70)	$96.13_{\pm 0.41}$ (3.86)	$88.94_{\pm 0.21}$ (0.60)	$87.32_{\pm 0.15}$ (1.09)	37.58
IU	$53.95_{\pm 1.24}$ (2.32)	$57.48_{\pm 0.17}$ (0.38)	$75.88_{\pm 1.16}$ (0.65)	$76.73_{\pm 0.74}$ (0.59)	$99.68_{\pm 0.11}$ (0.32)	$99.67_{\pm 0.05}$ (0.32)	$88.93_{\pm 0.10}$ (0.61)	$88.28_{\pm 0.14}$ (0.13)	3.11
Random data forgetting (all classes)									
Retrain	$5.41_{\pm 0.11}$	$6.77_{\pm 0.23}$	$13.12_{\pm 0.14}$	$14.17_{\pm 0.18}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$94.42_{\pm 0.09}$	$93.33_{\pm 0.12}$	42.15
FT	$6.83_{\pm 0.51}$ (1.42)	$5.97_{\pm 0.57}$ (0.80)	$14.97_{\pm 0.62}$ (1.85)	$13.36_{\pm 0.59}$ (0.81)	$96.61_{\pm 0.25}$ (3.39)	$96.99_{\pm 0.31}$ (3.01)	$92.13_{\pm 0.26}$ (2.29)	$92.29_{\pm 0.31}$ (1.04)	2.33
GA	$7.54_{\pm 0.29}$ (2.13)	$5.62_{\pm 0.46}$ (1.15)	$10.04_{\pm 0.31}$ (3.08)	$11.76_{\pm 0.52}$ (2.41)	$93.31_{\pm 0.04}$ (6.69)	$95.44_{\pm 0.11}$ (4.56)	$89.28_{\pm 0.07}$ (5.14)	$89.26_{\pm 0.15}$ (4.07)	0.31
FF	$7.84_{\pm 0.71}$ (2.43)	$8.16_{\pm 0.67}$ (1.39)	$9.52_{\pm 0.43}$ (3.60)	$10.80_{\pm 0.37}$ (3.37)	$92.05_{\pm 0.16}$ (7.95)	$92.29_{\pm 0.24}$ (7.71)	$88.10_{\pm 0.19}$ (6.32)	$87.79_{\pm 0.23}$ (5.54)	38.24
IU	$2.03_{\pm 0.43}$ (3.38)	$6.51_{\pm 0.52}$ (0.26)	$5.07_{\pm 0.74}$ (8.05)	$11.93_{\pm 0.68}$ (2.41)	$98.26_{\pm 0.29}$ (1.74)	$94.94_{\pm 0.31}$ (5.06)	$91.33_{\pm 0.22}$ (3.09)	$88.74_{\pm 0.42}$ (4.59)	3.22

[41], SVHN [43], and ImageNet [44]) and an alternative model architecture (VGG-16 [45]) can be found in Appendix 3.4. Across all the aforementioned datasets and model architectures, our experiments consistently show that model sparsification can effectively reduce the gap between approximate unlearning and exact unlearning.

Unlearning and pruning setups. We will focus on three unlearning scenarios mentioned in Sec. 2, *class-wise forgetting*, *random data forgetting (per class)*, and *random data forgetting (all classes)*. Unless specified otherwise, the class-wise forgetting will be the default setting.

In the '*prune first, then unlearn*' paradigm, we will focus on approximate unlearning methods (FT, GA, FF, and IU) shown in Tab. 1 when applying to sparse models. We implement these methods following their official repositories. However, it is worth noting that the implementation of FF [12] modifies the model architecture in class-wise forgetting, *i.e.*, removes the prediction head of the class to be scrubbed. By contrast, other methods keep the model architecture intact during unlearning. Also, we choose OMP as the default pruning method, as justified in Fig. 4. In the '*sparsity-aware unlearning*' paradigm, the sparsity-promoting regularization parameter γ in (3) is set to $\gamma = 10^{-4}$. This is found by a line search over $[10^{-5}, 10^{-1}]$ across tradeoffs between testing accuracy and unlearning accuracy. Further, we implement AO-sparse MU by choosing the pruning ratio $p\% = 20\%$ per iteration. We refer readers to Appendix 3.2 for more training details.

Evaluation metrics. We evaluate the unlearning performance following Tab. 1. Recall that UA and MIA-Efficacy depict the *efficacy* of MU, RA reflects the *fidelity* of MU, and TA and RTE characterize the *generalization ability* and the *computation efficiency* of an unlearning method. We implement MIA (membership inference attack) using the prediction confidence-based attack method [46, 47], whose effectiveness has been justified in [48] compared to other methods. We refer readers to Appendix 3.3 for more implementation details.

5.2 Experiment results

Model sparsity improves approximate unlearning. In **Tab. 2**, we study the impact of model sparsity on the performance of various MU methods in the 'prune first, then unlearn' paradigm. The performance of the exact (Retrain) method is also provided for comparison. Note that the better performance of approximate unlearning corresponds to the smaller performance gap with the gold-standard retrained model (Retrain).

First, given an approximate unlearning method (FT, GA, FF, or IU), we consistently observe that model sparsity improves UA and MIA-Efficacy (*i.e.*, the efficacy of approximate unlearning) without much performance loss in RA (*i.e.*, fidelity). In particular, the performance gap between each approximate unlearning method and Retrain reduces as the model becomes sparser (see the '95% sparsity' column vs. the 'dense' column). Note that the performance gap against Retrain is highlighted

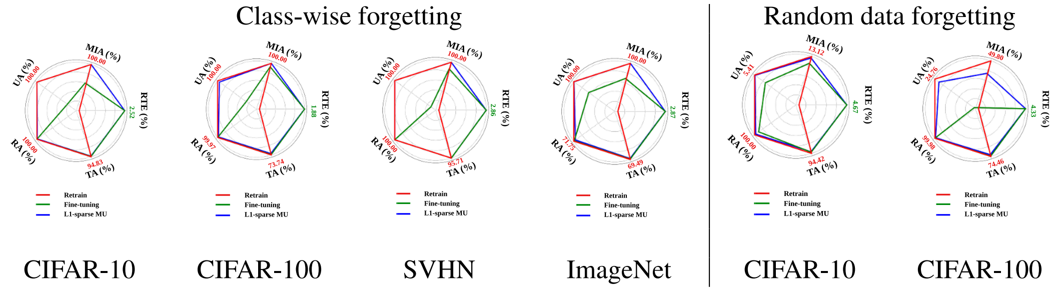

Figure 5: Sparsity-aware unlearning considering class-wise forgetting. **Left**: Unlearning accuracy vs. sparsity. **Right**: MIA-Efficacy vs. sparsity.

in $(\cdot)$ for each approximate unlearning. We also observe that Retrain on the 95%-sparsity model encounters a 3% TA drop. Yet, from the perspective of approximate unlearning, this drop brings in a more significant improvement in UA and MIA-Efficacy when model sparsity is promoted. Let us take FT (the simplest unlearning method) for class-wise forgetting as an example. As the model sparsity reaches 95%, we obtain 51% UA improvement and 8% MIA-Efficacy improvement. Furthermore, FT and IU on the 95%-sparsity model can better preserve TA compared to other approximate unlearning methods.

Second, existing approximate unlearning methods have different pros and cons. Let us focus on the regime of 95% sparsity. We observe that FT typically yields the best RA and TA, which has a tradeoff with its unlearning efficacy (UA and MIA-Efficacy). Moreover, GA yields the worst RA since it is most loosely connected with the remaining dataset $\mathcal{D}_\mathrm{r}$. FF becomes ineffective when scrubbing random data points compared to its class-wise unlearning performance. Furthermore, IU causes a TA drop but yields the smallest gap with exact unlearning across diverse metrics under the 95% model sparsity. We refer readers to Appendix 3.4 for more dataset results.

Efficacy of sparsity-aware unlearning. In **Tab. A5**, we present the performance of proposed sparsity-aware unlearning methods (*i.e.*, ℓ_1-sparse MU and AO-sparse MU). To justify the effectiveness of our proposal, we implement ℓ_1-sparse MU and AO-sparse MU following the objective of FT, the approximate unlearning method with the largest efficacy gap against exact unlearning on the original dense model as shown in Tab. 2. For comparison, the performance of Retrain, FT, and IU is also provided.

As shown in **Tab. A5**, ℓ_1-sparse MU improves the efficacy of unlearning (in terms of UA and MIA-Efficacy) over IU, and only has a quite small gap with Retrain even if the model considered for unlearning is dense (without ever pruning). This is because ℓ_1-sparse MU imposes a sparse regularization in (3) to penalize the model weights during unlearning. The downside of ℓ_1-sparse MU is its loss in RA and TA compared to IU and Retrain. However, enforcing its model sparsity seems beneficial to fix this issue, as supported by RA and TA of ℓ_1-sparse MU under 95% model sparsity.

Furthermore, we note that AO-sparse MU reduces to FT on dense model (*i.e.*, there exists no alternating optimization between unlearning and pruning when $p\% = 0$). Thus, AO-sparse MU only remains effective in the sparse regime ($p > 0$) and outperforms ℓ_1-sparse MU in general, particularly in RA and TA. More results can be found in Fig. A1. We also refer readers to Appendix 3.4 for more unlearning scenarios.

Model sparsity benefits privacy of MU for 'free'.

It was recently shown in [27, 28] that model sparsification helps protect data privacy, in terms of defense against MIA used to infer training data information from a learned model. Inspired by the above, we ask if sparsity can also bring the privacy benefit to an unlearned model, evaluated by the MIA rate on the remaining dataset $\mathcal{D}_\mathrm{r}$ (that we term **MIA-Privacy**). This is different from MIA-Efficacy, which reflects the efficacy of scrubbing $\mathcal{D}_\mathrm{f}$, *i.e.*, correctly deciding $\mathcal{D}_\mathrm{f}$ is not in the training set of the unlearned model. In contrast, MIA-Privacy characterizes the *privacy* of the unlearned model about $\mathcal{D}_\mathrm{r}$. A *lower* MIA-Privacy implies *less* information leakage.

Fig. 6 shows MIA-Privacy of unlearned models versus the sparsity ratio applied to different unlearning methods in the 'prune first, then unlearn' paradigm. As we can see, MIA-Privacy decreases as the sparsity increases. This suggests the improved privacy of unlearning on sparse models.

Moreover, we observe that approximate unlearning outperforms exact unlearning (Retrain) in privacy preservation of $\mathcal{D}_\mathrm{r}$. This is because Retrain is conducted over $\mathcal{D}_\mathrm{r}$ from scratch, leading to the strongest dependence on $\mathcal{D}_\mathrm{r}$ than other unlearning methods. Another interesting observation is that IU and GA yield a much smaller MIA-Privacy than other approximate unlearning methods. The rationale behind that is IU and GA have a weaker correlation with $\mathcal{D}_\mathrm{r}$ during unlearning. Specifically, the unlearning loss of IU only involves the forgetting data influence weights, *i.e.*, $(\mathbf{1}/N - \mathbf{w})$ in (1). Similarly, GA only performs gradient ascent over $\mathcal{D}_\mathrm{f}$, with the least dependence on $\mathcal{D}_\mathrm{r}$.

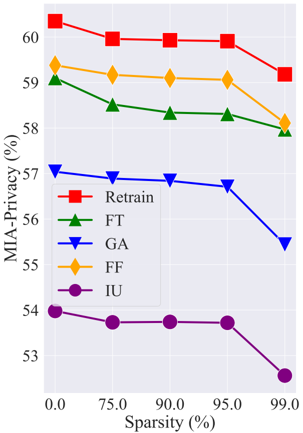

Figure 6: Privacy on $\mathcal{D}_\mathrm{r}$ (MIA-Privacy) using different unlearning methods vs. model sparsity.

A use case study: MU for Trojan model cleanse. We next present an application of MU to remove the influence of poisoned backdoor data from a learned model, following the backdoor attack setup in [49], where an adversary manipulates a small portion of training data (known as the poisoning ratio) by injecting a backdoor trigger (*e.g.*, a small patch or sticker on images) and/or modifying data labels towards a targeted incorrect label. The trained model is called *Trojan model*, which causes the backdoor-designated incorrect prediction if the trigger is present at testing. Otherwise, it behaves normally. That is, training over the poisoned data set will enforce a spurious correlation between the Trojan trigger and the model prediction, so that the former serves as a 'backdoor' for the trained model.

We then regard FT-based MU as a defensive method to scrub the harmful influence of poisoned training data in the model's prediction, with a similar motivation as [50]. We assume that the set of poisoned data points is known *a priori*, *e.g.*, via Trojan trigger detection [51]. We focus on the effect of MU on backdoor attack success rate (**ASR**) and standard accuracy (**SA**) of a Trojan model. **Fig. 7** shows ASR and SA of the Trojan model (with poisoning ratio 10%) and its unlearned version using the simplest FT method against model sparsity. As we can see, the Trojan model maintains 100% ASR and a similar SA across different model sparsity levels. By contrast, FT-based unlearning can reduce ASR without inducing much SA loss. Such a defensive advantage becomes more significant when sparsity reaches 90%. Thus, proposed unlearning gives an effective backdoor defense.

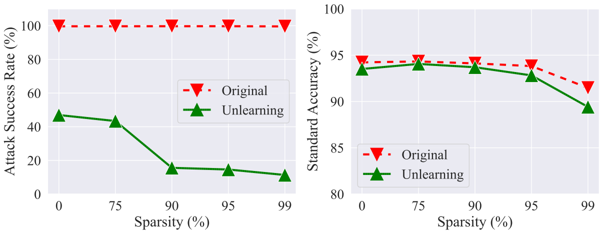

Figure 7: Performance of Trojan model cleanse via proposed unlearning vs. model sparsity, where 'Original' refers to the original Trojan model. **Left**: ASR vs. model sparsity. **Right**: SA vs. model sparsity. Each marker represents the mean value over 10 independent trials.

Additional results. Please refer to Appendix 3.

6 Related Work

Machine unlearning. In addition to exact and approximate unlearning methods as we have reviewed in Sec. 2, there exists other literature aiming to develop the probabilistic notion of unlearning [52–56], in particular through the lens of differential privacy (DP) [57]. Although DP enables unlearning with provable error guarantees, they typically require strong model and algorithmic assumptions and could lack effectiveness when facing practical adversaries, *e.g.*, membership inference attacks. Indeed, evaluating MU is far from trivial [8, 9, 11]. Furthermore, the attention on MU has also been raised in different learning paradigms, *e.g.*, federated learning [29, 58], graph neural network [59, 60], and adversarial ML [61, 62].

Understanding data influence. The majority of MU studies are motivated by data privacy. Yet, they also closely relate to another line of research on understanding data influence in ML. For example, the influence function approach [33] has been used as an algorithmic backbone of many unlearning methods [6, 10]. From the viewpoint of data influence, MU has been used in the use case of adversarial defense against data poisoning backdoor attacks [50]. Beyond unlearning, evaluation of data influence has also been studied in fair learning [63, 64], transfer learning [65], and dataset pruning [66, 67].

Model pruning. The deployment constraints on *e.g.*, computation, energy, and memory necessitate the pruning of today's ML models, *i.e.*, promoting their weight sparsity. The vast majority of existing works [13–19] focus on developing model pruning methods that can strike a graceful balance between model's generalization and sparsity. In particular, the existence of LTH (lottery ticket hypothesis) [15] demonstrates the feasibility of co-improving the model's generalization and efficiency (in terms of sparsity) [38, 68–70]. In addition to generalization, model sparsity achieved by pruning can also be leveraged to improve other performance metrics, such as robustness [20–22], model explanation [25, 26], and privacy [27, 28, 71, 72].

7 Conclusion

In this work, we advance the method of machine unlearning through a novel viewpoint: model sparsification, achieved by weight pruning. We show in both theory and practice that model sparsity plays a foundational and crucial role in closing the gap between exact unlearning and existing approximate unlearning methods. Inspired by that, we propose two new unlearning paradigms, 'prune first, then unlearn' and 'sparsity-aware unlearn', which can significantly improve the efficacy of approximate unlearning. We demonstrate the effectiveness of our findings and proposals in extensive experiments across different unlearning setups.

References

[1] Yinzhi Cao and Junfeng Yang, "Towards making systems forget with machine unlearning," in *2015 IEEE Symposium on Security and Privacy*. IEEE, 2015, pp. 463–480.

[2] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot, "Machine unlearning," in *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2021, pp. 141–159.

[3] Thanh Tam Nguyen, Thanh Trung Huynh, Phi Le Nguyen, Alan Wee-Chung Liew, Hongzhi Yin, and Quoc Viet Hung Nguyen, "A survey of machine unlearning," *arXiv preprint arXiv:2209.02299*, 2022.

[4] Jeffrey Rosen, "The right to be forgotten," *Stan. L. Rev. Online*, vol. 64, pp. 88, 2011.

[5] Chris Jay Hoofnagle, Bart van der Sloot, and Frederik Zuiderveen Borgesius, "The european union general data protection regulation: what it is and what it means," *Information & Communications Technology Law*, vol. 28, no. 1, pp. 65–98, 2019.

[6] Alexander Warnecke, Lukas Pirch, Christian Wressnegger, and Konrad Rieck, "Machine unlearning of features and labels," *arXiv preprint arXiv:2108.11577*, 2021.

[7] Laura Graves, Vineel Nagisetty, and Vijay Ganesh, "Amnesiac machine learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, vol. 35, pp. 11516–11524.

[8] Anvith Thudi, Gabriel Deza, Varun Chandrasekaran, and Nicolas Papernot, "Unrolling sgd: Understanding factors influencing machine unlearning," *arXiv preprint arXiv:2109.13398*, 2021.

[9] Alexander Becker and Thomas Liebig, "Evaluating machine unlearning via epistemic uncertainty," *arXiv preprint arXiv:2208.10836*, 2022.

[10] Zachary Izzo, Mary Anne Smart, Kamalika Chaudhuri, and James Zou, "Approximate data deletion from machine learning models," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 2008–2016.

[11] Anvith Thudi, Hengrui Jia, Ilia Shumailov, and Nicolas Papernot, "On the necessity of auditable algorithmic definitions for machine unlearning," in *31st USENIX Security Symposium (USENIX Security 22)*, 2022, pp. 4007–4022.

[12] Aditya Golatkar, Alessandro Achille, and Stefano Soatto, "Eternal sunshine of the spotless net: Selective forgetting in deep networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9304–9312.

[13] Song Han, Huizi Mao, and William J Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[14] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Michael Carbin, and Zhangyang Wang, "The lottery tickets hypothesis for supervised and self-supervised pre-training in computer vision models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 16306–16316.

[15] Jonathan Frankle and Michael Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," *arXiv preprint arXiv:1803.03635*, 2018.

[16] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin, "Linear mode connectivity and the lottery ticket hypothesis," in *International Conference on Machine Learning*. PMLR, 2020, pp. 3259–3269.

[17] Xiaolong Ma, Geng Yuan, Xuan Shen, Tianlong Chen, Xuxi Chen, Xiaohan Chen, Ning Liu, Minghai Qin, Sijia Liu, Zhangyang Wang, et al., "Sanity checks for lottery tickets: Does your winning ticket really win the jackpot?," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12749–12760, 2021.

[18] Yihua Zhang, Yuguang Yao, Parikshit Ram, Pu Zhao, Tianlong Chen, Mingyi Hong, Yanzhi Wang, and Sijia Liu, "Advancing model pruning via bi-level optimization," in *Advances in Neural Information Processing Systems*, 2022.

[19] Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag, "What is the state of neural network pruning?," *Proceedings of machine learning and systems*, vol. 2, pp. 129–146, 2020.

[20] Vikash Sehwag, Shiqi Wang, Prateek Mittal, and Suman Jana, "Hydra: Pruning adversarially robust neural networks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 19655–19666, 2020.

[21] Tianlong Chen, Zhenyu Zhang, Yihua Zhang, Shiyu Chang, Sijia Liu, and Zhangyang Wang, "Quarantine: Sparsity can uncover the trojan attack trigger for free," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 598–609.

[22] James Diffenderfer, Brian Bartoldson, Shreya Chaganti, Jize Zhang, and Bhavya Kailkhura, "A winning hand: Compressing deep networks can improve out-of-distribution robustness," *Advances in Neural Information Processing Systems*, vol. 34, pp. 664–676, 2021.

[23] Samuil Stoychev and Hatice Gunes, "The effect of model compression on fairness in facial expression recognition," *arXiv preprint arXiv:2201.01709*, 2022.

[24] Guangxuan Xu and Qingyuan Hu, "Can model compression improve nlp fairness," *arXiv preprint arXiv:2201.08542*, 2022.

[25] Eric Wong, Shibani Santurkar, and Aleksander Madry, "Leveraging sparse linear layers for debuggable deep networks," in *International Conference on Machine Learning*. PMLR, 2021, pp. 11205–11216.

[26] Tianlong Chen, Zhenyu Zhang, Jun Wu, Randy Huang, Sijia Liu, Shiyu Chang, and Zhangyang Wang, "Can you win everything with a lottery ticket?," *Transactions of Machine Learning Research*, 2022.

[27] Yangsibo Huang, Yushan Su, Sachin Ravi, Zhao Song, Sanjeev Arora, and Kai Li, "Privacy-preserving learning via deep net pruning," *arXiv preprint arXiv:2003.01876*, 2020.

[28] Yijue Wang, Chenghong Wang, Zigeng Wang, Shanglin Zhou, Hang Liu, Jinbo Bi, Caiwen Ding, and Sanguthevar Rajasekaran, "Against membership inference attack: Pruning is all you need," *arXiv preprint arXiv:2008.13578*, 2020.

[29] Junxiao Wang, Song Guo, Xin Xie, and Heng Qi, "Federated unlearning via class-discriminative pruning," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 622–632.

[30] Jingwen Ye, Yifang Fu, Jie Song, Xingyi Yang, Songhua Liu, Xin Jin, Mingli Song, and Xinchao Wang, "Learning with recoverable forgetting," in *European Conference on Computer Vision*. Springer, 2022, pp. 87–103.

[31] Sara Hooker, Aaron Courville, Gregory Clark, Yann Dauphin, and Andrea Frome, "What do compressed deep neural networks forget?," *arXiv preprint arXiv:1911.05248*, 2019.

[32] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter, "Continual lifelong learning with neural networks: A review," *Neural Networks*, vol. 113, pp. 54–71, 2019.

[33] Pang Wei Koh and Percy Liang, "Understanding black-box predictions via influence functions," in *International conference on machine learning*. PMLR, 2017, pp. 1885–1894.

[34] R Dennis Cook and Sanford Weisberg, *Residuals and influence in regression*, New York: Chapman and Hall, 1982.

[35] Sidak Pal Singh and Dan Alistarh, "Woodfisher: Efficient second-order approximation for neural network compression," *Advances in Neural Information Processing Systems*, vol. 33, pp. 18098–18109, 2020.

[36] Ga Wu, Masoud Hashemi, and Christopher Srinivasa, “Puma: Performance unchanged model augmentation for training data removal,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022, pp. 8675–8682.

[37] Kongyang Chen, Yao Huang, and Yiwen Wang, “Machine unlearning via gan,” *arXiv preprint arXiv:2111.11869*, 2021.

[38] Hidenori Tanaka, Daniel Kunin, Daniel L Yamins, and Surya Ganguli, “Pruning neural networks without any data by iteratively conserving synaptic flow,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 6377–6389, 2020.

[39] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M Roy, and Michael Carbin, “Pruning neural networks at initialization: Why are we missing the mark?,” *arXiv preprint arXiv:2009.08576*, 2020.

[40] Francis Bach, Rodolphe Jenatton, Julien Mairal, Guillaume Obozinski, et al., “Optimization with sparsity-inducing penalties,” *Foundations and Trends® in Machine Learning*, vol. 4, no. 1, pp. 1–106, 2012.

[41] Alex Krizhevsky, Geoffrey Hinton, et al., “Learning multiple layers of features from tiny images,” 2009.

[42] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[43] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng, “Reading digits in natural images with unsupervised feature learning,” 2011.

[44] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[45] Karen Simonyan and Andrew Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[46] Liwei Song, Reza Shokri, and Prateek Mittal, “Privacy risks of securing machine learning models against adversarial examples,” in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 241–257.

[47] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha, “Privacy risk in machine learning: Analyzing the connection to overfitting,” in *2018 IEEE 31st computer security foundations symposium (CSF)*. IEEE, 2018, pp. 268–282.

[48] Liwei Song and Prateek Mittal, “Systematic evaluation of privacy risks of machine learning models,” *arXiv preprint arXiv:2003.10595*, 2020.

[49] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg, “Badnets: Identifying vulnerabilities in the machine learning model supply chain,” *arXiv preprint arXiv:1708.06733*, 2017.

[50] Yang Liu, Mingyuan Fan, Cen Chen, Ximeng Liu, Zhuo Ma, Li Wang, and Jianfeng Ma, “Backdoor defense with machine unlearning,” *arXiv preprint arXiv:2201.09538*, 2022.

[51] Ren Wang, Gaoyuan Zhang, Sijia Liu, Pin-Yu Chen, Jinjun Xiong, and Meng Wang, “Practical detection of trojan neural networks: Data-limited and data-free cases,” in *European Conference on Computer Vision*. Springer, 2020, pp. 222–238.

[52] Antonio Ginart, Melody Guan, Gregory Valiant, and James Y Zou, “Making ai forget you: Data deletion in machine learning,” *Advances in neural information processing systems*, vol. 32, 2019.

[53] Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten, “Certified data removal from machine learning models,” *arXiv preprint arXiv:1911.03030*, 2019.

[54] Seth Neel, Aaron Roth, and Saeed Sharifi-Malvajerdi, "Descent-to-delete: Gradient-based methods for machine unlearning," in *Algorithmic Learning Theory*. PMLR, 2021, pp. 931–962.

[55] Enayat Ullah, Tung Mai, Anup Rao, Ryan A Rossi, and Raman Arora, "Machine unlearning via algorithmic stability," in *Conference on Learning Theory*. PMLR, 2021, pp. 4126–4142.

[56] Ayush Sekhari, Jayadev Acharya, Gautam Kamath, and Ananda Theertha Suresh, "Remember what you want to forget: Algorithms for machine unlearning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 18075–18086, 2021.

[57] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor, "Our data, ourselves: Privacy via distributed noise generation," in *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 2006, pp. 486–503.

[58] 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," *arXiv preprint arXiv:2203.07320*, 2022.

[59] Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, Mathias Humbert, and Yang Zhang, "Graph unlearning," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022, pp. 499–513.

[60] Eli Chien, Chao Pan, and Olgica Milenkovic, "Certified graph unlearning," *arXiv preprint arXiv:2206.09140*, 2022.

[61] Neil G Marchant, Benjamin IP Rubinstein, and Scott Alfeld, "Hard to forget: Poisoning attacks on certified machine unlearning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022, pp. 7691–7700.

[62] Jimmy Z Di, Jack Douglas, Jayadev Acharya, Gautam Kamath, and Ayush Sekhari, "Hidden poison: Machine unlearning enables camouflaged poisoning attacks," in *NeurIPS ML Safety Workshop*, 2022.

[63] Prasanna Sattigeri, Soumya Ghosh, Inkit Padhi, Pierre Dognin, and Kush R. Varshney, "Fair infinitesimal jackknife: Mitigating the influence of biased training data points without refitting," in *Advances in Neural Information Processing Systems*, 2022.

[64] Jialu Wang, Xin Eric Wang, and Yang Liu, "Understanding instance-level impact of fairness constraints," in *International Conference on Machine Learning*. PMLR, 2022, pp. 23114–23130.

[65] Saachi Jain, Hadi Salman, Alaa Khaddaj, Eric Wong, Sung Min Park, and Aleksander Madry, "A data-based perspective on transfer learning," *arXiv preprint arXiv:2207.05739*, 2022.

[66] Zalán Borsos, Mojmir Mutny, and Andreas Krause, "Coresets via bilevel optimization for continual learning and streaming," *Advances in Neural Information Processing Systems*, vol. 33, pp. 14879–14890, 2020.

[67] Shuo Yang, Zeke Xie, Hanyu Peng, Min Xu, Mingming Sun, and Ping Li, "Dataset pruning: Reducing training data by examining generalization influence," *arXiv preprint arXiv:2205.09329*, 2022.

[68] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell, "Rethinking the value of network pruning," *arXiv preprint arXiv:1810.05270*, 2018.

[69] Chaoqi Wang, Guodong Zhang, and Roger Grosse, "Picking winning tickets before training by preserving gradient flow," *arXiv preprint arXiv:2002.07376*, 2020.

[70] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip HS Torr, "Snip: Single-shot network pruning based on connection sensitivity," *arXiv preprint arXiv:1810.02340*, 2018.

[71] Zelun Luo, Daniel J Wu, Ehsan Adeli, and Li Fei-Fei, "Scalable differential privacy with sparse network finetuning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 5059–5068.

[72] Yifan Gong, Zheng Zhan, Zhengang Li, Wei Niu, Xiaolong Ma, Wenhao Wang, Bin Ren, Caiwen Ding, Xue Lin, Xiaolin Xu, et al., "A privacy-preserving-oriented dnn pruning and mobile acceleration framework," in *Proceedings of the 2020 on Great Lakes Symposium on VLSI*, 2020, pp. 119–124.

[73] Stephen Gould, Basura Fernando, Anoop Cherian, Peter Anderson, Rodrigo Santa Cruz, and Edison Guo, "On differentiating parameterized argmin and argmax problems with application to bi-level optimization," *arXiv preprint arXiv:1607.05447*, 2016.

1 Proof of Proposition 1

Recap the definition of model update $\Delta(\mathbf{w})$ in (1) and $\boldsymbol{\theta}_{\mathrm{o}} = \boldsymbol{\theta}(\mathbf{1}/N)$, we approximate $\Delta(\mathbf{w})$ by the first-order Taylor expansion of $\boldsymbol{\theta}(\mathbf{w})$ at $\mathbf{w} = \mathbf{1}/N$. This leads to

$$\Delta(\mathbf{w}) = \boldsymbol{\theta}(\mathbf{w}) - \boldsymbol{\theta}(\mathbf{1}/N) \approx \left.\frac{d\boldsymbol{\theta}(\mathbf{w})}{d\mathbf{w}}\right|_{\mathbf{w}=\mathbf{1}/N} (\mathbf{w} - \mathbf{1}/N), \tag{A1}$$

where $\frac{d\boldsymbol{\theta}(\mathbf{w})}{d\mathbf{w}} \in \mathbb{R}^{M \times N}$, and recall that $M = |\boldsymbol{\theta}_{\mathrm{o}}|$ is the number of model parameters. The gradient $\frac{d\boldsymbol{\theta}(\mathbf{w})}{d\mathbf{w}}$ is known as implicit gradient [73] since it is defined through the solution of the optimization problem $\boldsymbol{\theta}(\mathbf{w}) = \arg\min_{\boldsymbol{\theta}} L(\mathbf{w}, \boldsymbol{\theta})$, where recall that $L(\mathbf{w}, \boldsymbol{\theta}) = \sum_{i=1}^{N} [w_i \ell_i(\boldsymbol{\theta}, \mathbf{z}_i)]$. By the stationary condition of $\boldsymbol{\theta}(\mathbf{w})$, we obtain

$$\nabla_{\boldsymbol{\theta}} L(\mathbf{w}, \boldsymbol{\theta}(\mathbf{w})) = \mathbf{0}. \tag{A2}$$

Next, we take the derivative of (A2) w.r.t. $\mathbf{w}$ based on the implicit function theorem [73] assuming that $\boldsymbol{\theta}(\mathbf{w})$ is the unique solution to minimizing L. This leads to

$$\left[\frac{d\boldsymbol{\theta}(\mathbf{w})}{d\mathbf{w}}\right]^T \left[\nabla_{\boldsymbol{\theta},\boldsymbol{\theta}} L(\mathbf{w}, \boldsymbol{\theta})|_{\boldsymbol{\theta}=\boldsymbol{\theta}(\mathbf{w})}\right] + \nabla_{\mathbf{w},\boldsymbol{\theta}} L(\mathbf{w}, \boldsymbol{\theta}(\mathbf{w})) = \mathbf{0}, \tag{A3}$$

where $\nabla_{\mathbf{a},\mathbf{b}} = \nabla_{\mathbf{a}} \nabla_{\mathbf{b}} \in \mathbb{R}^{|\mathbf{a}| \times |\mathbf{b}|}$ is the second-order partial derivative. Therefore,

$$\frac{d\boldsymbol{\theta}(\mathbf{w})}{d\mathbf{w}} = -\left[\nabla_{\boldsymbol{\theta},\boldsymbol{\theta}} L(\mathbf{w}, \boldsymbol{\theta}(\mathbf{w}))\right]^{-1} \nabla_{\mathbf{w},\boldsymbol{\theta}} L(\mathbf{w}, \boldsymbol{\theta}(\mathbf{w}))^T, \tag{A4}$$

where $\nabla_{\mathbf{w},\boldsymbol{\theta}} L(\mathbf{w}, \boldsymbol{\theta}(\mathbf{w}))$ can be expanded as

$$\nabla_{\mathbf{w},\boldsymbol{\theta}} L(\mathbf{w}, \boldsymbol{\theta}(\mathbf{w})) = \nabla_{\mathbf{w}} \nabla_{\boldsymbol{\theta}} \sum_{i=1}^{N} [w_i \ell_i(\boldsymbol{\theta}(\mathbf{w}), \mathbf{z}_i)] = \nabla_{\mathbf{w}} \sum_{i=1}^{N} [w_i \nabla_{\boldsymbol{\theta}} \ell_i(\boldsymbol{\theta}(\mathbf{w}), \mathbf{z}_i)] = \begin{bmatrix} \nabla_{\boldsymbol{\theta}} \ell_i(\boldsymbol{\theta}(\mathbf{w}), \mathbf{z}_1)^T \\ \nabla_{\boldsymbol{\theta}} \ell_i(\boldsymbol{\theta}(\mathbf{w}), \mathbf{z}_2)^T \\ \vdots \\ \nabla_{\boldsymbol{\theta}} \ell_i(\boldsymbol{\theta}(\mathbf{w}), \mathbf{z}_N)^T \end{bmatrix}. \tag{A5}$$

Based on (A4) and (A5), we obtain the closed-form of implicit gradient at $\mathbf{w} = \mathbf{1}/N$:

$$\begin{aligned} \frac{d\boldsymbol{\theta}(\mathbf{w})}{d\mathbf{w}} |_{\mathbf{w}=\mathbf{1}/N} &= -\left[\nabla_{\boldsymbol{\theta},\boldsymbol{\theta}} L(\mathbf{1}/N, \boldsymbol{\theta}(\mathbf{1}/N))\right]^{-1} \left[\nabla_{\boldsymbol{\theta}} \ell_i(\boldsymbol{\theta}(\mathbf{1}/N), \mathbf{z}_1) \quad \ldots \quad \nabla_{\boldsymbol{\theta}} \ell_i(\boldsymbol{\theta}(\mathbf{1}/N), \mathbf{z}_N)\right] \\ &= -\mathbf{H}^{-1} \left[\nabla_{\boldsymbol{\theta}} \ell_i(\boldsymbol{\theta}(\mathbf{1}/N), \mathbf{z}_1) \quad \ldots \quad \nabla_{\boldsymbol{\theta}} \ell_i(\boldsymbol{\theta}(\mathbf{1}/N), \mathbf{z}_N)\right], \end{aligned} \tag{A6}$$

where $\mathbf{H} = \nabla_{\boldsymbol{\theta},\boldsymbol{\theta}} L(\mathbf{1}/N, \boldsymbol{\theta}(\mathbf{1}/N))$.

Substituting (A6) into (A1), we obtain

$$\begin{aligned} \Delta(\mathbf{w}) &\approx -\mathbf{H}^{-1} \left[\nabla_{\boldsymbol{\theta}} \ell_i(\boldsymbol{\theta}(\mathbf{1}/N), \mathbf{z}_1) \quad \ldots \quad \nabla_{\boldsymbol{\theta}} \ell_i(\boldsymbol{\theta}(\mathbf{1}/N), \mathbf{z}_N)\right] (\mathbf{w} - \mathbf{1}/N) \\ &= -\mathbf{H}^{-1} \sum_{i=1}^{N} [(w_i - 1/N) \nabla_{\boldsymbol{\theta}} \ell_i(\boldsymbol{\theta}(\mathbf{1}/N), \mathbf{z}_i)] \\ &= \mathbf{H}^{-1} \nabla_{\boldsymbol{\theta}} L(\mathbf{1}/N - \mathbf{w}, \boldsymbol{\theta}_{\mathrm{o}}), \end{aligned} \tag{A7}$$

where the last equality holds by the definition of $L(\mathbf{w}, \boldsymbol{\theta}) = \sum_{i=1}^{N} [w_i \ell_i(\boldsymbol{\theta}, \mathbf{z}_i)]$.

The proof is now complete.

2 Proof of Proposition 2

The proof follows [8, Sec. 5], with the additional condition that the model is **sparse** encoded by a pre-fixed (binary) pruning mask $\mathbf{m}$, namely, $\boldsymbol{\theta}' := \mathbf{m} \odot \boldsymbol{\theta}$. Then, based on [8, Eq. 5], the model updated by SGD yields

$$\boldsymbol{\theta}'_t \approx \boldsymbol{\theta}'_0 - \eta \mathbf{m} \odot \sum_{i=1}^{t-1} \nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}'_0, \hat{\mathbf{z}}_i) + \mathbf{m} \odot (\sum_{i=1}^{t-1} f(i)), \tag{A8}$$

where $\boldsymbol{\theta}'_0 = \mathbf{m} \odot \boldsymbol{\theta}_0$ is the model initialization when using SGD-based sparse training, $\{\hat{\mathbf{z}}_i\}$ is the sequence of stochastic data samples, t is the number of training iterations, η is the learning rate, and $f(i)$ is defined recursively as

$$f(i) = -\eta \nabla^2_{\boldsymbol{\theta},\boldsymbol{\theta}} \ell(\boldsymbol{\theta}'_0, \hat{\mathbf{z}}_i) \left(-\eta \sum_{j=0}^{i-1} \mathbf{m} \odot \nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}'_0, \hat{\mathbf{z}}_j) + \sum_{j=0}^{i-1} (\mathbf{m} \odot f(j)) \right), \tag{A9}$$

with $f(0) = 0$. Inspired by the second term of (A8), to unlearn the data sample $\hat{\mathbf{z}}_i$, we will have to add back the first-order gradients under $\hat{\mathbf{z}}_i$. This corresponds to the GA-based approximate unlearning method. Yet, this approximate unlearning introduces an unlearning error, given by the last term of (A8)

$$\mathbf{e}_{\mathbf{m}}(\boldsymbol{\theta}_0, \{\hat{\mathbf{z}}_i\}, t, \eta) := \mathbf{m} \odot (\sum_{i=1}^{t-1} f(i)). \tag{A10}$$

Next, if we interpret the mask $\mathbf{m}$ as a diagonal matrix $\operatorname{diag}(\mathbf{m})$ with 0's and 1's along its diagonal based on $\mathbf{m}$, we can then express the sparse model $\mathbf{m} \odot \boldsymbol{\theta}$ as $\operatorname{diag}(\mathbf{m})\boldsymbol{\theta}$. Similar to [8, Eq. 9], we can derive a bound on the unlearning error (A10) by ignoring the terms other than those with η^2 in $f(i)$ (*i.e.*, (A9)). This yields

$$\begin{aligned}
e(\mathbf{m}) = \|\mathbf{e}_{\mathbf{m}}(\boldsymbol{\theta}_0, \{\hat{\mathbf{z}}_i\}, t, \eta)\|_2 &= \left\| \mathbf{m} \odot (\sum_{i=1}^{t-1} f(i)) \right\|_2 \\
&\approx \eta^2 \left\| \operatorname{diag}(\mathbf{m}) \sum_{i=1}^{t-1} \nabla^2_{\boldsymbol{\theta},\boldsymbol{\theta}} \ell(\boldsymbol{\theta}'_0, \hat{\mathbf{z}}_i) \sum_{j=0}^{i-1} \mathbf{m} \odot \nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}'_0, \hat{\mathbf{z}}_j) \right\|_2 && \text{(A11)} \\
&\leq \eta^2 \sum_{i=1}^{t-1} \left\| \operatorname{diag}(\mathbf{m}) \nabla^2_{\boldsymbol{\theta},\boldsymbol{\theta}} \ell(\boldsymbol{\theta}'_0, \hat{\mathbf{z}}_i) \sum_{j=0}^{i-1} \mathbf{m} \odot \nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}'_0, \hat{\mathbf{z}}_j) \right\|_2 && \text{(Triangle inequality)} \\
&\leq \eta^2 \sum_{i=1}^{t-1} \left\| \operatorname{diag}(\mathbf{m}) \nabla^2_{\boldsymbol{\theta},\boldsymbol{\theta}} \ell(\boldsymbol{\theta}'_0, \hat{\mathbf{z}}_i) \right\| \left\| \sum_{j=0}^{i-1} \mathbf{m} \odot \nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}'_0, \hat{\mathbf{z}}_j) \right\|_2 && \text{(A12)} \\
&\lesssim \eta^2 \sum_{i=1}^{t-1} \left\| \operatorname{diag}(\mathbf{m}) \nabla^2_{\boldsymbol{\theta},\boldsymbol{\theta}} \ell(\boldsymbol{\theta}'_0, \hat{\mathbf{z}}_i) \right\| \frac{i}{t} \left\| \boldsymbol{\theta}'_t - \boldsymbol{\theta}'_0 \right\|_2 \\
&\leq \eta^2 \sigma(\mathbf{m}) \left\| \mathbf{m} \odot (\boldsymbol{\theta}_t - \boldsymbol{\theta}_0) \right\|_2 \frac{1}{t} \frac{t-1}{2} t = \frac{\eta^2}{2} (t-1) \|\mathbf{m} \odot (\boldsymbol{\theta}_t - \boldsymbol{\theta}_0)\|_2 \sigma(\mathbf{m}), && \text{(A13)}
\end{aligned}$$

where the second last inequality holds given the fact that $\sum_{j=0}^{i-1} \mathbf{m} \odot \nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}'_0, \hat{\mathbf{z}}_j)$ can be approximated by its expectation $\frac{i(\boldsymbol{\theta}'_t - \boldsymbol{\theta}'_0)}{t}$ [8, Eq. 7], and $\sigma(\mathbf{m}) := \max_j \{\sigma_j(\nabla^2_{\boldsymbol{\theta},\boldsymbol{\theta}} \ell), \text{if } m_j \neq 0\}$, *i.e.*, the largest eigenvalue among the dimensions left intact by the binary mask $\mathbf{m}$. The above suggests that the unlearning error might be large if $\mathbf{m} = \mathbf{1}$ (no pruning). Based on (A13), we can then readily obtain the big O notation in (2). This completes the proof.

3 Additional experimental details and results

3.1 Datasets and models

We summarize the datasets and model configurations in Table A1.

Table A1: Dataset and model setups.

Settings	CIFAR-10			SVHN	CIFAR-100	ImageNet
	ResNet-18	ResNet-20s	VGG-16	ResNet-18	ResNet-18	ResNet-18
Batch Size	128	128	128	128	128	1024

Table A2: Hyperparameters settings for pruning.

Experiments	CIFAR-10/CIFAR-100	SVHN	ImageNet
Training epochs	182	160	90
Rewinding epochs	8	8	5
Momentum	0.9	0.9	0.875
ℓ_2 regularization	$5e^{-4}$	$5e^{-4}$	$3.05e^{-5}$
Warm-up epochs	1(75 for VGG-16)	0	8

3.2 Detailed training and unlearning settings

Training details of pruning. For all pruning methods, including IMP [15], SynFlow [38], and OMP [17], we adopt the settings from the current SOTA implementations [17]; see a summary in Tab. A2. For IMP, OMP, and SynFlow, we adopt the step learning rate scheduler with a decay rate of 0.1 at 50% and 75% epochs. We adopt 0.1 as the initial learning rate for all pruning methods.

Training details of unlearning. For all datasets and model architecture, we adopt 10 epochs for FT, and 5 epochs for GA method. The learning rate for FT and GA are carefully tuned between $[10^{-5}, 0.1]$ for each dataset and model architecture. In particular, we adopt 0.01 as the learning rate for FT method and 10^{-4} for GA on the CIFAR-10 dataset (ResNet-18) at different sparsity levels. By default, we choose SGD as the optimizer for the FT and GA methods. As for FF method, we perform a greedy search for hyperparameter tuning [12] between 10^{-8} and 10^{-6}.

3.3 Detailed metric settings

Details of MIA implementations. MIA is implemented using the prediction confidence-based attack method [48]. There are mainly two phases during its computation: **(1) training phase**, and **(2) testing phase**. To train an MIA model, we first sample a balanced dataset from the remaining dataset ($\mathcal{D}_\mathrm{r}$) and the test dataset (different from the forgetting dataset $\mathcal{D}_\mathrm{f}$) to train the MIA predictor. The learned MIA is then used for MU evaluation in its testing phase. To evaluate the performance of MU, MIA-Efficacy is obtained by applying the learned MIA predictor to the unlearned model ($\boldsymbol{\theta}_\mathrm{u}$) on the forgetting dataset ($\mathcal{D}_\mathrm{f}$). Our objective is to find out how many samples in $\mathcal{D}_\mathrm{f}$ can be correctly predicted as non-training samples by the MIA model against $\boldsymbol{\theta}_\mathrm{u}$. The formal definition of MIA-Efficacy is then given by:

$$\text{MIA-Efficacy} = \frac{TN}{|\mathcal{D}_\mathrm{f}|},$$

where TN refers to the true negatives predicted by our MIA predictor, *i.e.*, the number of the forgetting samples predicted as non-training examples, and $|\mathcal{D}_\mathrm{f}|$ refers to the size of the forgetting dataset. As described above, MIA-Efficacy leverages the privacy attack to justify how good the unlearning performance could be.

3.4 Additional experiment results

Overall performance on various datasets and model architectures. In Tab. A3, we show that sparsity can reduce the performance gap between approximate unlearning and exact unlearning on various datasets and different model architectures in the class-wise forgetting unlearning scenario. As we can see, model sparsity improves UA and MIA-Efficacy without much performance loss in RA and TA across various datasets, which is consistent with results shown in Table 2. Table A3 also shows the consistent observations on the VGG-16 model architecture.

Overall performance on ImageNet. To demonstrate the effectiveness of our methods on a larger dataset, we conducted additional experiments on ImageNet with other experiment settings consistent with the class-wise forgetting in Table 2. As we can see from Table A4, sparsity reduces the performance gap between exact unlearning (Retrain) and the approximate unlearning methods (FT

Table A3: Performance overview of MU vs. sparsity on different datasets considering class-wise forgetting. The content format follows Table 2.

MU	UA		MIA-Efficacy		RA		TA		RTE
	DENSE	**95% Sparsity**	DENSE	**95% Sparsity**	DENSE	**95% Sparsity**	DENSE	**95% Sparsity**	(min)
CIFAR-100, ResNet-18									
Retrain	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$99.97_{\pm 0.01}$	$96.68_{\pm 0.15}$	$73.74_{\pm 0.19}$	$69.49_{\pm 0.41}$	48.45
FT	$26.45_{\pm 6.29}$ (73.55)	$73.63_{\pm 5.06}$ (26.37)	$92.44_{\pm 5.93}$ (7.56)	$98.88_{\pm 4.32}$ (15.60)	$99.86_{\pm 0.04}$(0.11)	$97.72_{\pm 0.47}$ (1.04)	$74.08_{\pm 0.23}$ (0.74)	$71.37_{\pm 0.18}$ (3.00)	1.88
GA	$81.47_{\pm 0.32}$(18.53)	$99.01_{\pm 0.01}$ (0.99)	$93.47_{\pm 4.56}$ (6.53)	$100.00_{\pm 0.00}$ (0.00)	$90.33_{\pm 1.71}$ (9.64)	$80.45_{\pm 0.78}$ (16.23)	$64.94_{\pm 0.74}$ (8.80)	$60.99_{\pm 0.14}$ (8.50)	0.21
IU	$84.12_{\pm 0.34}$ (15.88)	$99.78_{\pm 0.01}$ (0.22)	$98.44_{\pm 0.45}$ (1.56)	$99.33_{\pm 0.00}$ (0.67)	$96.23_{\pm 0.02}$ (3.74)	$95.45_{\pm 0.17}$ (1.23)	$71.24_{\pm 0.22}$ (2.50)	$70.79_{\pm 0.11}$ (0.95)	1.30
SVHN, ResNet-18									
Retrain	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$95.71_{\pm 0.12}$	$94.95_{\pm 0.05}$	42.84
FT	$11.48_{\pm 8.12}$ (88.52)	$51.93_{\pm 19.62}$ (48.07)	$86.12_{\pm 9.62}$ (13.88)	$99.42_{\pm 0.51}$ (0.58)	$100.00_{\pm 0.00}$ (0.00)	$99.00_{\pm 0.00}$ (1.00)	$95.99_{\pm 0.07}$ (0.28)	$95.89_{\pm 0.02}$ (0.94)	2.86
GA	$83.87_{\pm 0.19}$ (16.13)	$86.52_{\pm 0.11}$ (13.48)	$99.97_{\pm 0.02}$ (0.03)	$100.00_{\pm 0.00}$ (0.00)	$99.60_{\pm 0.15}$ (0.40)	$98.37_{\pm 0.11}$ (1.63)	$95.27_{\pm 0.02}$ (0.44)	$93.42_{\pm 0.07}$ (1.53)	0.28
IU	$95.11_{\pm 0.02}$ (4.89)	$100.00_{\pm 0.00}$ (0.00)	$99.89_{\pm 0.04}$ (0.11)	$100.00_{\pm 0.00}$(0.00)	$100.00_{\pm 0.00}$ (0.00)	$99.85_{\pm 0.02}$ (0.15)	$95.70_{\pm 0.09}$ (0.01)	$94.90_{\pm 0.04}$ (0.05)	3.19
CIFAR-10, VGG-16									
Retrain	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.01}$	$99.97_{\pm 0.00}$	$94.83_{\pm 0.10}$	$92.93_{\pm 0.06}$	30.38
FT	$28.00_{\pm 8.16}$ (72.00)	$34.94_{\pm 5.37}$ (65.06)	$63.23_{\pm 17.68}$ (36.77)	$68.02_{\pm 12.03}$ (31.98)	$99.87_{\pm 0.05}$(0.13)	$99.60_{\pm 0.08}$ (0.37)	$92.80_{\pm 1.28}$ (2.03)	$92.96_{\pm 0.85}$ (0.03)	1.89
GA	$77.51_{\pm 3.47}$ (22.49)	$83.93_{\pm 2.14}$ (16.07)	$80.13_{\pm 4.27}$ (19.87)	$88.04_{\pm 3.18}$ (11.96)	$96.09_{\pm 0.13}$(3.91)	$97.33_{\pm 0.08}$ (2.64)	$88.80_{\pm 1.33}$ (6.03)	$89.95_{\pm 0.78}$ (2.98)	0.27
IU	$88.58_{\pm 0.86}$ (11.42)	$98.78_{\pm 0.44}$ (1.22)	$92.27_{\pm 1.14}$ (7.73)	$99.91_{\pm 0.05}$ (0.09)	$96.89_{\pm 0.27}$(3.11)	$93.18_{\pm 0.28}$ (6.79)	$89.81_{\pm 1.01}$ (5.02)	$87.45_{\pm 0.81}$ (5.48)	2.51

and GA). The results are consistent with our existing observations in Table 2. Note that the 83% model sparsity (ImageNet, ResNet-18) is used to preserve the TA performance after one-shot magnitude (OMP) pruning.

Table A4: Performance overview of MU vs. sparsity on ImageNet considering class-wise forgetting. The content format follows Table 2.

MU	UA		MIA-Efficacy		RA		TA		RTE
	DENSE	**83% Sparsity**	DENSE	**83% Sparsity**	DENSE	**83% Sparsity**	DENSE	**83% Sparsity**	(hours)
ImageNet									
Retrain	100.00	100.00	100.00	100.00	71.75	69.18	69.49	68.86	26.18
FT	63.60 (36.40)	74.66 (25.34)	68.61 (31.39)	81.43 (18.57)	72.45 (0.70)	69.36 (0.18)	69.80 (0.31)	68.77 (0.09)	2.87
GA	85.10 (14.90)	90.21 (9.79)	87.46 (12.54)	94.25 (5.75)	65.93 (5.82)	62.94 (6.24)	64.62 (4.87)	64.65 (4.21)	0.01

More results vs. sparsity ratio of sparsity-aware unlearning. Fig. A1 shows the performance of proposed sparsity-aware unlearning methods at different sparsity levels of class-wise forgetting. We can observe that ℓ_1-sparse MU outperforms IU at multiple sparsity levels. We also remark that the unlearning performance of AO-sparse MU improves as the sparsity increases, and reaches the best unlearning efficacy when the sparsity approaches 90%.

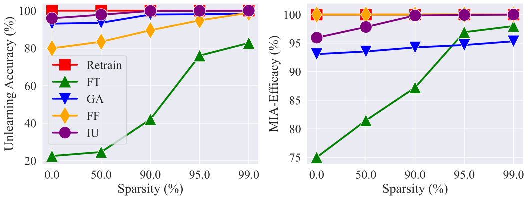

Figure A1: Sparsity-aware unlearning considering class-wise forgetting. **Left**: Unlearning accuracy vs. sparsity. **Right**: MIA-Efficacy vs. sparsity.

Table A5: Performance overview of sparsity-aware MU vs. Retrain, FT and IU considering class-wise forgetting and random data forgetting (all classes), where the table format and setup are consistent with Table 2. N/A represents the unlearning scenario over a dense model that AO-sparse MU cannot be directly applied.

MU	UA		MIA-Efficacy		RA		TA		RTE
	DENSE	**95% Sparsity**	DENSE	**95% Sparsity**	DENSE	**95% Sparsity**	DENSE	**95% Sparsity**	(min)
Class-wise Forgetting									
Retrain	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$99.99_{\pm 0.01}$	$94.83_{\pm 0.11}$	$91.80_{\pm 0.89}$	43.23
FT	$22.53_{\pm 8.16}$ (77.47)	$73.64_{\pm 9.46}$ (26.36)	$75.00_{\pm 14.68}$ (25.00)	$83.02_{\pm 16.33}$ (16.98)	$99.87_{\pm 0.04}$ (0.13)	$99.87_{\pm 0.05}$ (0.12)	$94.31_{\pm 0.19}$ (0.52)	$94.32_{\pm 0.12}$ (2.52)	2.52
IU	$87.82_{\pm 2.15}$ (12.18)	$99.47_{\pm 0.15}$ (0.53)	$95.96_{\pm 0.21}$ (4.04)	$99.93_{\pm 0.04}$ (0.07)	$97.98_{\pm 0.21}$ (2.02)	$97.24_{\pm 0.13}$ (2.75)	$91.42_{\pm 0.21}$ (3.41)	$90.76_{\pm 0.18}$ (1.04)	3.23
ℓ_1-sparse MU	$100.00_{\pm 0.00}$ (0.00)	$100.00_{\pm 0.00}$ (0.00)	$100.00_{\pm 0.00}$ (0.00)	$100.00_{\pm 0.00}$ ((0.00)	$98.99_{\pm 1.57}$ (1.01)	$95.74_{\pm 0.13}$ (4.26)	$93.40_{\pm 1.39}$ (1.02)	$88.97_{\pm 1.00}$ (2.83)	5.84
AO-sparse MU	N/A	$99.80_{\pm 0.19}$(0.20)	N/A	$100.00_{\pm 0.00}$(0.00)	N/A	$99.86_{\pm 0.05}$ (0.13)	N/A	$94.55_{\pm 0.11}$(4.25)	2.86
Random data forgetting (all classes)									
Retrain	$5.41_{\pm 0.11}$	$6.77_{\pm 0.23}$	$13.12_{\pm 0.14}$	$14.17_{\pm 0.18}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$94.42_{\pm 0.09}$	$93.33_{\pm 0.12}$	42.15
FT	$6.83_{\pm 0.51}$ (1.42)	$5.97_{\pm 0.57}$ (0.80)	$14.97_{\pm 0.62}$ (1.85)	$13.36_{\pm 0.59}$ (0.81)	$96.61_{\pm 0.25}$ (3.39)	$96.99_{\pm 0.31}$ (3.01)	$92.13_{\pm 0.26}$ (2.29)	$92.29_{\pm 0.31}$ (1.04)	2.33
IU	$2.03_{\pm 0.43}$ (3.38)	$6.51_{\pm 0.52}$ (0.26)	$5.07_{\pm 0.74}$ (8.05)	$11.93_{\pm 0.68}$ (2.41)	$98.26_{\pm 0.29}$ (1.74)	$94.94_{\pm 0.31}$ (5.06)	$91.33_{\pm 0.22}$ (3.09)	$88.74_{\pm 0.42}$ (4.59)	3.22
ℓ_1-sparse MU	$6.60_{\pm 0.35}$ (1.19)	$5.71_{\pm 0.35}$ (1.06)	$14.64_{\pm 0.35}$ (1.52)	$13.71_{\pm 0.46}$ (1.03)	$96.51_{\pm 0.50}$ (3.49)	$96.97_{\pm 3.03}$ (1.78)	$92.66_{\pm 0.35}$ (1.76)	$92.60_{\pm 0.34}$ (0.73)	4.99
AO-sparse MU	N/A	$5.80_{\pm 0.35}$ (0.97)	N/A	$14.67_{\pm 0.43}$ (0.50)	N/A	$98.22_{\pm 0.39}$ (1.78)	N/A	$92.33_{\pm 0.34}$ (1.00)	3.34