Minimally Invasive Machine Unlearning via Posterior Control

Anonymous authors
Paper under double-blind review

Abstract

Approximate Machine Unlearning (MU) methods typically forget specific data by modifying model parameters or learning data-dependent augmentations. However, each unlearning request requires a dedicated optimization process, resulting in high computational overhead and cumulative degradation of model performance over time. Recent approaches have proposed manipulating a small subset of neural activations as a more targeted alternative, yet these methods still rely on on-demand searches for relevant parameters and remain computationally expensive. We propose Minimally Invasive Machine Unlearning (MIMU), a posterior control–based MU framework that enables unlearning without inference-time optimization. MIMU introduces a parameterized control module that maps unlearning requests to binary masking policies via forward propagation. The resulting masks identify the most influential parameters associated with the forget set, allowing their predictive influence to be effectively removed without additional optimization when an unlearning request is issued. Experimental results across multiple benchmarks demonstrate that MIMU achieves competitive unlearning performance while preserving the generalization ability and prediction patterns of the original model, offering a favorable trade-off between unlearning efficiency and effectiveness.

1 Introduction

Many advanced Machine Unlearning (MU) methods adopt approximate unlearning objectives Nguyen et al. (2025) that update the model by jointly minimizing changes in predictive error on the retained training data while maximizing the error on the data to be forgotten. Although these approaches can effectively erase memorization of target data, they often (1) unintentionally alter the model's overall behavior Cooper et al. (2025) and (2) require inference-time optimization upon receiving an unlearning request Wang et al. (2024). As a result, they can degrade the model's generalization ability and substantially increase the computational cost of frequent unlearning operations.

To mitigate the performance degradation caused by frequent unlearning operations, recent research has shifted focus toward methods that operate on a limited number of neural parameters or activations. These techniques aim to reduce the computational footprint and minimize disruption to the model's learned representation by identifying and updating only the weights most relevant to the data points to be forgotten. For instance, Mehta et al. (2022) uses conditional independence analysis to identify semantically relevant parameters for each data point. Similarly, Malekmohammadi et al. (2025) proposes selecting the sharpest parameters, those with the largest Hessian diagonal values, for unlearning updates. Other approaches, such as Torkzadehmahani et al. (2024), reset and fine-tune only the most critical parameters identified through memorization analysis. Layer-specific methods like Gogineni & Nadimi (2024) combine pruning and update tracking to localize unlearning to particular layers, while Cha et al. (2023) employ adversarial relabeling and weight importance to guide parameter changes. More recently, Li et al. (2025) introduced a patching framework with certifiable guarantees that modifies localized parameter regions. These works show that MU can be achieved through targeted parameter intervention, though they often suffer from high computational complexity due to the need of instance-specific parameter search or fine-tuning.

We propose Minimally Invasive Machine Unlearning (MIMU), the first posterior control–based machine unlearning framework that eliminates inference-time optimization for individual unlearning requests, thereby

significantly reducing both computational cost and model invasiveness. Inspired by posterior control theory Li & Rush (2020), MIMU learns a parameterized probabilistic control module that maps an unlearning request (expressed as a forget set) to a sparse parameter mask identifying the parameters most responsible for the target predictions. Unlike iterative, instance-specific optimization methods Deo et al. (2025) or layer-wise heuristic approaches Xu et al. (2024); Dong et al. (2017), MIMU trains and caches its control module prior to receiving any unlearning request, enabling highly efficient processing at inference time. Extensive experiments demonstrate that a well-trained MIMU control module reliably identifies high-influence parameter subsets for arbitrary unlearning requests. By directly masking these parameters, MIMU effectively removes predictive capability on the forget set while preserving both predictive performance and decision patterns on retained data, achieving competitive results against state-of-the-art unlearning baselines. Further analysis shows that, in addition to its exceptional inference-time efficiency, MIMU exhibits several desirable properties: (1) it preserves prediction pattern of model on retain data points (evidenced by saliency map analysis), and (2) it drives predictive confidence on the forget set toward high predictive entropy, rather than confidently misclassifying forgotten samples into incorrect classes that are observed in many existing methods.

2 Preliminary

MU methods are divided into two classifications, exact and approximate unlearning Neel et al. (2021); Thudi et al. (2022b). Exact unlearning requires training the model such that retraining it without the forget data is more efficient than naively retraining the entire model from scratch. Early works on exact unlearning achieved this by exploiting additive (summation-based) training formulations, Cao & Yang (2015), or smart division of the data for targeted unlearning Bourtoule et al. (2020); Yan et al. (2022a). While standard exact machine unlearning (MU) methods are more efficient than full retraining, they can still suffer substantial computational costs or high storage overhead from maintaining and ensembling models trained on disjoint data shards Yan et al. (2022b). Approximate unlearning is a faster and more cost-efficient approach for removing the influence of specific data points. Indeed approximate unlearning methods Guo et al. (2023); Golatkar et al. (2020); Thudi et al. (2022a) Warnecke et al. (2023); Kurmanji et al. (2023); Li et al. (2024) succeed in time and storage efficiency but they require updates across all parameters within the model which can lead to deterioration of the model's generalization.

To preserve the computation efficiency and reduce damage on the unlearned model's generalization, recent developments Wu et al. (2022); Jia et al. (2024) have focused on targeting the parameters of the model that correspond to the data points to be unlearned. Mehta et al. (2022) perform MU on particular rows (or "slices") of layers within a neural network instead of the entire layer such that an expensive inverse Hessian adjustment can be performed piece by piece. Goel et al. (2023) "freezes" all but the last k layers of a DNN and offer two approaches, retraining those k layers from scratch or fine-tuning the k layers on the retain dataset. Focusing on the later layers allows the model to keep all the lower-level representations it has learned in hopes of preserving the model's generalization. Cha et al. (2023) measure a weight's importance with "memory aware synapses" Aljundi et al. (2018) and use that weight importance to penalize changes to weights that are not important to the forget set during finetuning. Malekmohammadi et al. (2025) adopt a layer-wise approach by computing the average Hessian for each layer and performing unlearning on the layer with the flattest curvature. The authors argue that layers associated with flatter loss landscapes are more difficult to unlearn; hence, successful unlearning in such layers is expected to be more resistant to relearning attacks. Fan et al. (2024) examined the saliency of model weights with respect to the forget set and mask weights whose gradients exceed a predefined threshold. Such weights are deemed more influential for the data to be unlearned and are therefore removed during the unlearning process. Ding et al. (2025) refined this approach to check for crossover importance for both the retain and forget dataset, advising to abstain from masking any neurons with too much importance across both sets. Torkzadehmahani et al. (2024) take an adapted approach to saliency and determine the most important weights for the forget set by the product of the magnitude of the weight and the gradient of the loss for the forget data set.

2.1 Objectives of Machine Unlearning

While most machine unlearning (MU) methods are motivated by privacy concerns, the community has converged on two closely related but distinct interpretations of MU. The original MU definition requires the unlearned model to behave indistinguishably from a model retrained from scratch Bourtoule et al. (2021), primarily to satisfy Right-to-be-Forgotten regulations found in legislation such as the General Data Protection Regulation (GDPR) in the European Union. However, evaluating unlearning effectiveness by direct comparison with a retrained model remains contentious Wichert & Sikdar (2024). In contrast, a more operational but implicit definition emphasizes explicitly suppressing predictive capability on the forget set while preserving generalization on the retain data Wu et al. (2022). This objective better aligns with the need of repairing deployed models (by removing training data that encode undesirable properties). Notably, although many existing methods are introduced under the first definition, their training objectives and loss formulations more closely reflect the second interpretation.

Our work aligns more closely with this second interpretation: preserving the model's predictive performance on the retain data and maximizing the degradation of the model's accuracy and confidence on the forget set. Nevertheless, in our experiments, we still include comparisons against fully retrained models, as is common practice in prior work Chundawat et al. (2022). Here, the minimal invasion emphasizes achieving unlearning with minimal perturbation of prediction patterns on the retain dataset to preserve overall model behavior as first priority and then effectively forgetting the target data as much as possible.

3 Minimally Invasive Machine Unlearning

Now, we present Minimally Invasive Machine Unlearning (MIMU), a control-based MU model framework to facilitate forget set predictability removal through dynamic weight masking. Given a trained predictive model $\hat{Y} = f_\theta(X)$ and a set of training data samples $D_{\text{train}} = \{(x, y)\}_m$, we aim to learn a masking function $Q_\phi(\cdot)$ that upon receiving an unlearning request defined by a forget set, produces a masking policy $\mathbf{m}_{\text{forget}} \in \mathbb{B}^{|\theta|}$. This policy identifies the subset of parameters in the predictive model that is most responsible for achieving effective unlearning. In other words, the mask generated should maximize the alignment between the original model's predictions in the forget set before and after applying the mask, so that:

$$|P_\theta(y_i|x_i) - P_{\mathbf{m}_{\text{forget}} \odot \theta}(y_i|x_i)| \leq \epsilon \quad \forall (x_i, y_i) \in D_{\text{forget}} \tag{1}$$

for some arbitrary small ϵ and satisfies the sparsity condition $||\mathbf{m}_{\text{forget}}||_0 = \delta$, with pre-defined sparsity target δ. Here, $\odot$ denotes element-wise multiplication. By masking out the identified parameters with $1 - \mathbf{m}_{\text{forget}} \odot \theta$, one can remove the predictability of the model on the forget set.

Departing from existing unlearning paradigms that update model parameters or optimize additional parameters for each unlearning request, MIMU does not directly alter the original predictive model. Instead, it treats each unlearning request as a model patching operation that adjusts model behavior through masking out small group of parameters.

3.1 Posterior Control for Optimal Mask Searching

The core idea of performing machine unlearning via a learned masking function naturally aligns with the objective of model control, in which a control policy is designed to steer a system toward a desired reference state (here, the removal of predictability on the forget set) while respecting resource and operational constraints. In our case, the masking function is a control module.

If we formalize the control module in a probabilistic setting, where the masking policy produced by the controller is modeled as a stochastic policy drawn from a parametric distribution, we can learn a parameterized control module by reducing KL-divergence with respect to an ideal but intractable masking policy P (or reference distribution).

$$\begin{aligned} &\arg\min_{\phi} KL\left[Q_\phi(\mathbf{m}_i|(x_i, y_i), \theta)||P(\mathbf{m}_i|(x_i, y_i), \theta)\right] \\ &\quad s.t. \quad ||\mathbf{m}_i||_0 = \delta, \end{aligned} \tag{2}$$

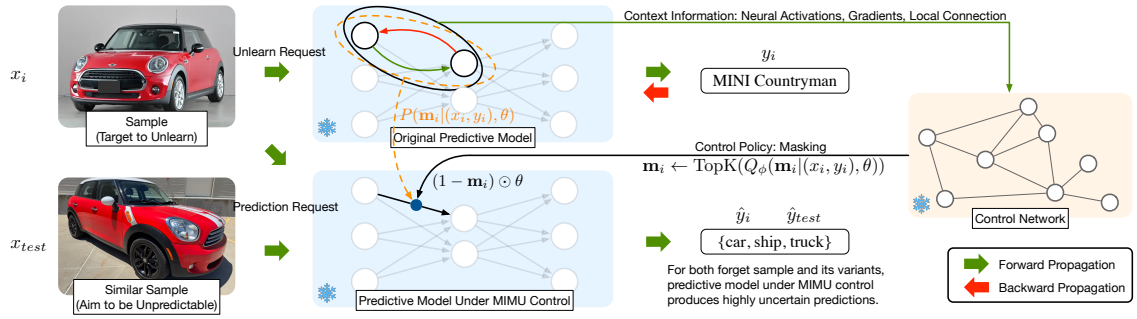


Figure 1: Illustrative figure of MIMU during inference time. Given a pre-trained parameterized control network $Q_\phi(\mathbf{m}_i|(x_i, y_i), \theta)$ that approximate the intractable masking distribution $P(\mathbf{m}_i|(x_i, y_i), \theta)$, the control network will conduct machine unlearning task by applying masked weights upon receiving unlearning request, thereby removing the model's predictive ability regarding similar input data without inference time optimization.

where the reference distribution P is defined as probability of masking given observed predictive model θ and data i

$$P(\mathbf{m}_i|(x_i, y_i), \theta) = \frac{P_{\mathbf{m}_i \odot \theta}(y_i|x_i)\cancel{P(x_i)}P(\mathbf{m}_i|\theta)}{P_\theta(y_i|x_i)\cancel{P(x_i)}}. \tag{3}$$

As a side note, some existing approaches Labach & Valaee (2020); Nguyen et al. (2022); Salehinejad & Valaee (2022), directly maximize the intractable distribution P with respect to $\mathbf{m}_i$ for each data point i as the Maximize a Posterior (MAP) objective. Unfortunately, the optimization has to be conducted through MCMC-like algorithms Chen et al. (2014); Ma et al. (2015) that iteratively search for the best mask, leading to significant computational cost.

By transforming Equation 2 with respect to the decomposition of the reference distribution (Equation 3), we instead look for modeling $\mathbf{m}_i$ as an parameterized distribution through the following training objective

$$\begin{aligned} &\arg\min_{\phi} \max_{\lambda} -E_{Q_\phi}[\log P_{\mathbf{m}_i \odot \theta}(y_i|x_i)] + \\ &KL[Q_\phi(\mathbf{m}_i|(x_i, y_i), \theta)||P(\mathbf{m}_i|\theta)] + \lambda(||\mathbf{m}_i||_0 - \delta), \end{aligned} \tag{4}$$

which is the combination of a cross entropy term, a KL regularization term, and a Lagrange multiplier term. The prior distribution $P(\mathbf{m}_i|\theta) \coloneqq P(\mathbf{m}|\theta)$ by default is a categorical distribution tied to the predictive model's parameter distribution. In particular, given the conclusion from previous study Jia et al. (2024), we set the prior distribution as

$$P(\mathbf{m}|\theta) = \frac{1}{\sum_{k'} ||\theta_{k'}||}||\theta_k|| \quad \forall k \in \{1 \cdots |\theta|\} \tag{5}$$

to reflect the intuition that larger weights account for larger influence on the model's prediction.

Given that both $Q_\phi(\mathbf{m}_i|(x_i, y_i), \theta)$ and $P(\mathbf{m}|\theta)$ are categorical distributions, the KL regularization term in Equation 4 can be minimized by reducing cross-entropy loss and encouraging diversity

$$\begin{aligned} &KL[Q_\phi(\mathbf{m}_i|(x_i, y_i), \theta)||P(\mathbf{m}|\theta)] \\ &= -\int_{\mathbf{m}_i} Q_\phi(\mathbf{m}_i|(x_i, y_i), \theta) \log P(\mathbf{m}|\theta) d\mathbf{m}_i \\ &\quad + \int_{\mathbf{m}_i} Q_\phi(\mathbf{m}_i|(x_i, y_i), \theta) \log Q_\phi(\mathbf{m}_i|(x_i, y_i), \theta) d\mathbf{m}_i. \end{aligned} \tag{6}$$

Optimizing Equation 4 is still hard due to the difficulty of enforcing the duality of the Lagrange multiplier term. Although implementing L1 regularization can effectively approximate the objective function, it often

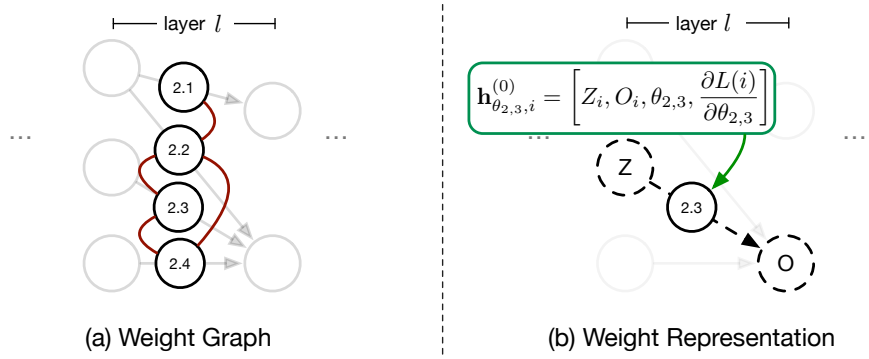


Figure 2: Illustrative figure to show the parameter graph construction on a layer of neural network. (a) a pair of parameters θ_k and $\theta_{k'}$ will have a connection, when $\theta_k \to v \wedge \theta_{k'} \to v$ or $v \to \theta_k \wedge v \to \theta_{k'}$ with shared node v. (b) the feature of each parameter θ_k consists of the surrounding information.

violates the sparsity constraint. To avoid optimization duality, we approximate the objective function by top-δ sampling during training so that

$$\mathbf{m}_i \sim \text{Top-}\delta(Q_\phi(\mathbf{m}_i|(x_i, y_i), \theta)), \tag{7}$$

where the number of positive samples ($m_{i,k} = 1$) can be enforced through $Q_\phi(m_{i,k}|(x_i, y_i), \theta)$ ranked top-δ filtering. With the controlled sampling, we now summarize the final implementable training objective as

$$\begin{aligned} \arg\min_{\phi} \sum_{(x_i,y_i)\sim D} & [-E_{\mathbf{m}_i \sim \text{Top-}\delta(Q_\phi)}[\log P_{\mathbf{m}_i \odot \theta}(y_i|x_i)] \\ & - E_{\mathbf{m}_i \sim \text{Top-}\delta(Q_\phi)}[\log P(\mathbf{m}|\theta)] + E_{\mathbf{m}_i \sim \text{Top-}\delta(Q_\phi)}[\log Q_\phi]], \end{aligned} \tag{8}$$

where we use Q_ϕ to denote $Q_\phi(\mathbf{m}_i|(x_i, y_i), \theta)$ for better clarity. Note that, while the parameter ϕ in the proposed distribution Q appears non-differentiable due to passing through a sampling step, we note the Gumbel reparameterization with a Top-K softmax filtering Jang et al. (2017); Maddison et al. (2017); Xie & Ermon (2021) can effectively smooth the path and make the objective trainable with respect to Q_ϕ.

3.2 Graph Convolution based Control Module

Modern neural predictive models often contain millions or even billions of parameters, making it impractical to design a control module $Q_\phi(m_{i,k}|(x_i, y_i), \theta)$ that jointly conditions on the input data and the full parameter set while producing a mask over all parameters in a single step. To address this scalability challenge, we restrict the scope of masking to a selected layer and instantiate the control module as a graph convolutional network (GCN).

Given a trained predictive model and a selected layer l, we represent the parameters of that layer as a graph $G = (V, E)$. Each node $v \in V$ corresponds to an individual weight parameter θ_k in the selected layer. An edge $e \in E$ connects a pair of parameters (θ_k, $\theta_{k'}$) if they are associated with the same neuron in the predictive model, either through incoming or outgoing connections, as shown in Figure 2.

The feature of each node θ_k with respect to each training data point (x_i, y_i) consists of 1) value of inbound activation $z(x_i)$ of layer l, 2) outbound activation $o(x_i)$, 3) weight value θ_k, and 4) derivative of the weight $\frac{\partial L(f_\theta(x_i), y_i)}{\partial \theta_k}$.

GCN estimates node embeddings through message passing such that

$$\mathbf{h}^{(t)}_{\theta_k,i} = \sigma\left(A^{(t)} \cdot \frac{1}{\mathcal{N}(\theta_k)} \sum_{\theta' \in \mathcal{N}(\theta_k)} \mathbf{h}^{(t-1)}_{\theta_{k'},i} + B^{(t)} \cdot \mathbf{h}^{(t-1)}_{k,i}\right), \tag{9}$$

where $A^{(t)}$ and $B^{(t)}$ are trainable parameters of the graph convolution network and $\mathbf{h}^{(0)}_{\theta_k,i}$ is the input feature θ_k of data point i. The output of each node from the graph convolution network is a probability of masking

$$Q_\phi(m_{i,k}|(x_i, y_i), \theta) = \text{Softmax}\left(W \cdot \mathbf{h}^{(T)}_{\theta_k,i} + b\right), \tag{10}$$

where the parameter group ϕ denotes all parameters of the GCN such that $\phi = \{\{A^{(t)}, B^{(t)}\}_{t\in\{1:T\}} \cup W\}$.

Although using a graph convolution-based masking function helps address local connectivity concerns, it can still struggle when scaling to model a layer with millions of parameters. Fortunately, prior work Fel et al. (2024); Dorszewski et al. (2025) has shown that the early layers of neural networks primarily act as generic feature extractors and contribute less directly to specific predictions. This insight allows us to reduce computational overhead by focusing the masking and unlearning process on the last one to two layers of the network, where they have the most direct influence on predictions but with far less number of parameters.

3.3 Unlearning via Filtered Posterior Control

Given the learned control module that obtains the important parameter subset for each training data point, we can process unlearning request for one specific training sample (x_i, y_i) by masking the minimal subset of model parameters that uniquely support its/their prediction.

While directly masking all top-δ parameters from control module as described in Equation 7 should allow us to push down the predictability of forget data point (x_i, y_i), it will inevitably hurt the model's generalization given the mask can destroy prediction pattern of some retain data points.

To reduce the risk of overshooting, we filter the masking suggestions from the control module by selecting a representative sample and applying a binary masking operation:

$$\tilde{\theta} \leftarrow \theta \odot (1 - \textit{filter}(\mathbf{m}_i, \kappa)),$$

where *filter*$(\cdot)$ denotes weighted filtering function that selects top $\kappa \ll \delta$ parameters to mask. In particular, we may stop masking when 1) the predictive confidence $f_{\tilde{\theta}}(x_i)[y_i]$ of forget sample drops below a predictive confidence threshold (meaning successful unlearning), 2) the predictive confidence of retain samples drops below the performance degradation tolerance, or 3) let κ to be a percentage of δ. In other words, κ can be dynamically selected for better unlearning outcome. In our experiments later, we enforce $\delta \leq 10\%$ of parameters in the selected layer of the target predictive model.

The above described unlearning strategy can be easily expand to set unlearning, where one may want to unlearn multiple training data at once. The only modification is to replace individual masking proposal $\mathbf{m}_i$ from the control module with an aggregated union of masks from all N forget set samples such that

$$\mathbf{m}_{\text{forget-set}} = \bigvee_{i=1}^{N} \mathbf{m}_i \tag{11}$$

In practice, when training samples in the forget set share common predictive patterns, we can subsample from the N samples to reduce computational cost. We also select a smaller κ so that $\kappa \cdot N \ll \delta$

Under

4 Experiments and Evaluations

Datasets and models We evaluate the proposed machine unlearning method, MIMU, against several unlearning algorithms on standard image classification benchmarks, including MNIST, Fashion-MNIST (FMNIST), CIFAR-10, and ImageNet100. For MNIST, we employ a multilayer perceptron (MLP) as the predictive model. For FMNIST, we use a convolutional neural network (CNN). Experiments on CIFAR-10 are conducted using the ResNet-18 architecture, while for ImageNet100 we adopt ResNet-50 (Appendix B).

Baselines We compare the proposed approach against a set of machine unlearning baselines, using *Retraining from scratch* (Retrained) as a reference. The baselines include Random Labelling (RL) Li & Ghosh (2023),

Table 1: Class-level unlearning performance across candidate algorithms in terms of retain and forget set accuracy. Results are averaged over 5 runs with standard deviation error bars.

Method	MNIST		FMNIST		CIFAR10		ImageNet100	
	Forget	Retain	Forget	Retain	Forget	Retain	Forget	Retain
Original	100.00 ± 0.00	100.00 ± 0.00	89.12 ± 0.00	93.39 ± 0.00	99.96 ± 0.00	99.87 ± 0.00	98.54 ± 0.00	92.57 ± 0.00
Retrained	0.00 ± 0.00	100.00 ± 0.00	0.00 ± 0.00	99.37 ± 0.52	0.00 ± 0.00	100.00 ± 0.00	0.00 ± 0.00	92.14 ± 0.04
Random Labels	17.79 ± 10.49	72.86 ± 6.15	54.24 ± 4.23	91.08 ± 0.53	2.55 ± 4.36	15.16 ± 09.65	95.58 ± 02.84	80.34 ± 01.86
Catastrophic Forgetting	99.79 ± 0.11	100.00 ± 0.00	30.24 ± 10.94	94.29 ± 0.34	0.11 ± 0.15	99.17 ± 0.41	13.32 ± 20.19	89.56 ± 0.55
Exact Unlearning	0.00 ± 0.00	98.91 ± 0.53	0.00 ± 0.00	81.65 ± 2.16	0.00 ± 0.00	99.56 ± 0.09	8.54 ± 8.29	90.66 ± 0.24
NegGrad+	0.00 ± 0.00	95.77 ± 0.35	0.00 ± 0.00	79.72 ± 0.44	0.00 ± 0.00	99.45 ± 0.14	0.00 ± 0.00	87.84 ± 0.40
L1-Sparse	0.00 ± 0.00	99.77 ± 0.01	0.00 ± 0.00	88.48 ± 0.21	0.00 ± 0.00	99.80 ± 0.02	0.08 ± 0.00	90.47 ± 0.03
Saliency Unlearning	0.18 ± 0.37	100.00 ± 0.00	0.00 ± 0.00	95.62 ± 0.16	0.00 ± 0.00	94.63 ± 0.39	0.00 ± 0.00	95.31 ± 0.47
SSD	23.24 ± 6.34	100.00 ± 0.00	3.90 ± 1.04	92.36 ± 0.62	4.03 ± 7.82	95.33 ± 1.67	0.00 ± 0.00	92.55 ± 0.01
SCRUB	0.00 ± 0.00	98.68 ± 0.01	0.54 ± 0.11	92.32 ± 0.02	0.00 ± 0.00	99.81 ± 0.00	0.00 ± 0.00	90.70 ± 0.03
MIMU	0.78 ± 1.32	98.78 ± 0.36	0.73 ± 0.52	92.55 ± 0.46	0.00 ± 0.00	99.64 ± 0.02	0.00 ± 0.00	90.72 ± 0.08

Table 2: Sub-class/random unlearning performance across candidate algorithms in terms of retain and forget set accuracy. Sub-classes are randomly selected from k-means clusters. Results are averaged over 5 runs with standard deviation error bars.

Method	MNIST		FMNIST		CIFAR10		ImageNet100	
	Forget	Retain	Forget	Retain	Forget	Retain	Forget	Retain
Original	100.00 ± 0.00	100.00 ± 0.00	76.10 ± 0.00	93.29 ± 0.00	99.88 ± 0.00	99.88 ± 0.00	97.58 ± 0.00	92.62 ± 0.00
Retrained	98.89 ± 0.11	100.00 ± 0.00	40.21 ± 9.87	99.36 ± 0.69	56.08 ± 1.95	99.99 ± 0.03	97.41 ± 0.38	92.22 ± 0.03
Random Labels	4.97 ± 0.98	90.40 ± 0.08	38.30 ± 6.660	89.13 ± 0.45	97.36 ± 2.34	94.38 ± 4.06	93.39 ± 5.66	86.31 ± 1.24
Catastrophic Forgetting	98.79 ± 0.91	98.84 ± 0.31	68.85 ± 9.97	92.91 ± 0.50	88.47 ± 9.81	99.09 ± 0.31	96.69 ± 1.55	89.41 ± 0.52
Exact Unlearning	99.07 ± 0.46	99.35 ± 0.09	30.09 ± 18.80	76.67 ± 1.70	98.60 ± 1.05	99.55 ± 0.11	97.02 ± 0.46	90.38 ± 0.52
NegGrad+	96.79 ± 0.27	99.97 ± 0.01	14.55 ± 0.47	91.04 ± 0.20	61.18 ± 0.18	99.44 ± 0.07	47.82 ± 1.68	89.05 ± 0.29
L1-Sparse	99.13 ± 0.06	99.17 ± 0.01	54.81 ± 0.81	92.35 ± 0.04	99.31 ± 0.14	99.85 ± 0.01	97.18 ± 0.00	90.56 ± 0.02
Saliency Unlearning	96.67 ± 2.25	100.00 ± 0.00	72.13 ± 21.32	94.09 ± 0.58	89.08 ± 3.85	96.87 ± 0.56	89.70 ± 5.95	95.23 ± 0.25
SSD	71.16 ± 15.74	99.15 ± 0.83	7.88 ± 7.16	63.11 ± 29.36	4.03 ± 7.82	95.33 ± 1.66	0.00 ± 0.00	91.80 ± 0.02
SCRUB	98.84 ± 0.04	98.40 ± 0.01	61.41 ± 0.38	90.07 ± 0.05	99.26 ± 0.00	99.85 ± 0.00	95.89 ± 0.18	90.85 ± 0.03
MIMU	70.52 ± 7.81	97.24 ± 0.76	66.76 ± 20.64	91.90 ± 0.23	73.74 ± 3.46	99.41 ± 0.05	90.00 ± 0.87	92.06 ± 0.08

Catastrophic Forgetting (CF@k) Goel et al. (2023) Exact Unlearning (EU@k) Goel et al. (2023) , L1-Sparse Jia et al. (2024), Saliency Unlearning (SalUn) Fan et al. (2024), Selective Synaptic Dampening (SSD) Foster et al. (2024), NegGrad+ Kurmanji et al. (2023), and SCRUB Kurmanji et al. (2023) (see Appendix 5). In particular, CF and EU are lightweight MU approaches that operate on a selected l-th layer of the model. In our experiments, we apply them to the second-to-last layer, consistent with the MIMU setting.

4.1 Overall Unlearning Performance

We next present the quantitative performance evaluation of the candidate unlearning algorithms. The evaluation covers multiple aspects: (1) accuracy on the retain and forget sets after unlearning, (2) prediction uncertainty (measured by entropy). It is worth noting that, while we include retraining from scratch as a reference, our goal is not to match the unlearned model to the retrained one. Instead, the objective is to remove the model's ability to predict the forget set while preserving its performance on the retain set.

Table 1 reports the prediction accuracy of unlearned models on the forget and retain datasets. The results show that MIMU achieves competitive unlearning performance compared to existing approaches that need inference-time optimization. In particular, we note MIMU significantly reduces the model's predictive ability on the forget set (with near 0% prediction accuracy) while effectively preserved model's performance on the retain dataset (with $\leq 2\%$ performance degradation) over all datasets in the experiments. It is the most

Table 3: Class-level unlearning performance in terms of post-unlearning prediction entropy. Results are averaged over 5 runs with standard deviation error bars.

Method	MNIST		FMNIST		CIFAR10		ImageNet100	
	Forget	Retain	Forget	Retain	Forget	Retain	Forget	Retain
Original	0.00 ± 0.00	0.00 ± 0.00	0.33 ± 0.00	0.19 ± 0.00	0.00 ± 0.00	0.01 ± 0.00	0.08 ± 0.00	0.28 ± 0.00
Retrained	0.26 ± 0.03	0.00 ± 0.00	0.35 ± 0.10	0.03 ± 0.01	0.34 ± 0.01	0.00 ± 0.00	2.01 ± 0.11	0.32 ± 0.00
Random Labels	2.13 ± 0.05	0.80 ± 0.12	1.22 ± 0.02	0.37 ± 0.01	2.27 ± 0.03	2.25 ± 0.01	2.06 ± 0.53	1.72 ± 0.20
Catastrophic Forgetting	0.01 ± 0.00	0.00 ± 0.00	0.76 ± 0.02	0.17 ± 0.01	0.66 ± 0.26	0.04 ± 0.01	1.46 ± 0.24	0.22 ± 0.00
Exact Unlearning	0.35 ± 0.11	0.02 ± 0.00	1.92 ± 0.06	1.45 ± 0.06	1.03 ± 0.31	0.05 ± 0.00	1.65 ± 0.09	0.22 ± 0.00
NegGrad+	0.00 ± 0.00	0.00 ± 0.00	0.01 ± 0.00	0.04 ± 0.00	0.19 ± 0.07	0.00 ± 0.00	0.04 ± 0.01	0.02 ± 0.00
L1-Sparse	1.00 ± 0.02	0.07 ± 0.00	1.61 ± 0.01	1.11 ± 0.00	1.20 ± 0.03	0.06 ± 0.00	3.10 ± 0.02	0.79 ± 0.00
Saliency Unlearning	2.21 ± 0.01	0.01 ± 0.00	0.62 ± 0.02	0.15 ± 0.00	0.40 ± 0.11	0.21 ± 0.02	1.21 ± 0.19	0.18 ± 0.01
SSD	0.55 ± 0.02	0.00 ± 0.00	0.62 ± 0.07	0.22 ± 0.00	0.81 ± 0.07	0.06 ± 0.01	2.33 ± 0.04	0.28 ± 0.00
SCRUB	0.96 ± 0.01	0.10 ± 0.00	1.22 ± 0.01	0.45 ± 0.00	0.93 ± 0.02	0.03 ± 0.00	3.27 ± 0.05	0.82 ± 0.01
MIMU	0.86 ± 0.05	0.03 ± 0.00	0.69 ± 0.06	0.25 ± 0.01	1.62 ± 0.02	0.02 ± 0.00	2.87 ± 0.03	0.37 ± 0.01

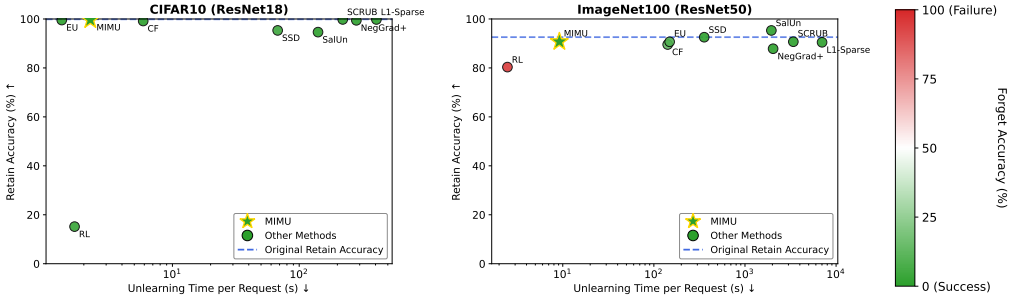


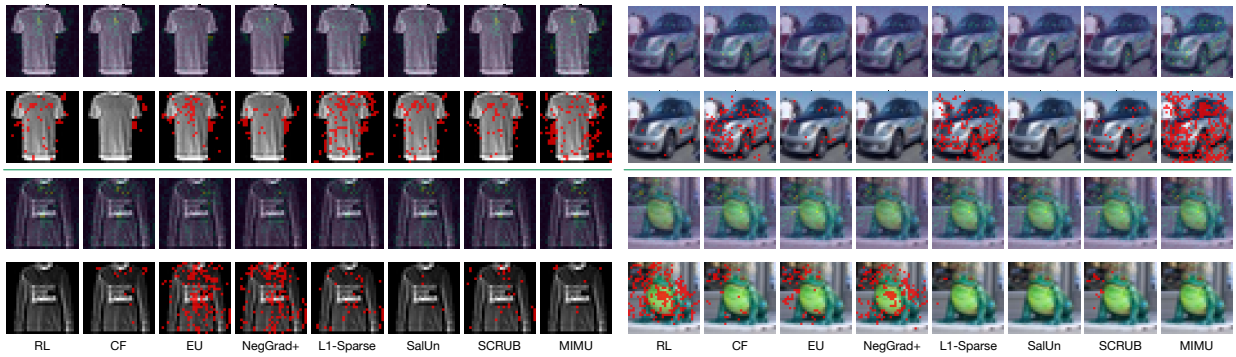
Figure 3: For every unlearning request, MIMU delivers the competitive results quickly. Other common MU techniques deliver similar performance but can cost significantly more in compute time. The graph above shows the Retain Accuracy in percentage on the Y axis, the Unlearning Time measured in wall-clock seconds on the X axis and the Forget Accuracy is measure from the hue of the marker, from red (failure - up to 100% retention) to green (success - down to 0% retention). Time values can be found in Table 6 in the Appendix.

stably performed MU algorithm compared to the baseline methods. Further examination of prediction confidence and uncertainty in Table 3 reveals that MIMU substantially increases the entropy of the forget set while maintaining low entropy for the retain set. This demonstrates that MIMU genuinely removes predictive capability rather than merely inducing incorrect predictions with high confidence.

Subclass-level unlearning is particularly challenging due to strong entanglement between the predictive patterns of the forget set and retain data points within the same class. As a result, many machine unlearning (MU) methods struggle to balance effective forgetting with performance preservation on retain data, often leading to two failure modes, either the unlearning process degrades the target model, or it fails to sufficiently weaken predictability on the forget set. Table 2 presents results for subclass-level unlearning. Under a fixed performance degradation tolerance, MIMU achieves the best trade-off between unlearning effectiveness and retain-set performance. In contrast, most baseline methods preserve (or even improve) retain-set accuracy at the cost of ineffective unlearning, highlighting the inherent trade-off in subclass-level unlearning objectives.

4.2 Unlearning Efficiency

We next evaluate the computational efficiency of unlearning algorithms by measuring the wall-clock processing time required to handle each unlearning request. Figure 3 summarizes the unlearning effectiveness and computational cost across all methods. The x-axis represents the average unlearning time per request,



(a) FMNIST (b) CIFAR10

Figure 4: Saliency map analysis before and after unlearning. The top row shows the saliency map of a forget sample after unlearning. The second row shows the saliency difference between before and after unlearning for the forget sample. The bottom two rows show the corresponding results for a retain sample. Red dots highlight regions with notable saliency differences, which implies prediction pattern shift of target model on the data.

while the y-axis reports the retain accuracy after unlearning. The color gradient indicates the forget accuracy, where green shade corresponds to lower forget accuracy (effective forgetting). The dashed horizontal line denotes the accuracy of the original model before unlearning. MIMU is highlighted with star markers, whereas competing methods are represented by circular markers.

As shown in Figure 3 across both CIFAR-10 and ImageNet-100, MIMU preserves retain accuracy close to that of the original model while simultaneously reducing forget accuracy to zero. Importantly, MIMU demonstrates significantly lower unlearning time per request compared to optimization-based approaches. The computational advantage of MIMU becomes more pronounced as model complexity grows. For example, on ImageNet-100 using a ResNet-50 backbone, methods that require iterative optimization during unlearning experience a substantial increase in runtime, whereas MIMU remains efficient. More detailed runtime comparisons are provided in Table 6. MIMU on CIFAR-10 and ImageNet-100 exhibits a substantial computational advantage over strong baselines , including NegGrad+, L1-Sparse, and SCRUB. While lightweight methods such as CF and EU achieve competitive execution times on smaller datasets, their computational complexity leads to significantly increased runtime when applied to larger models or datasets. Random Labeling (RL) is computationally inexpensive; however, this efficiency comes at the cost of effectiveness. As shown in Tables 1 and 2, RL often degrades generalization performance or fails to meaningfully suppress predictability on the forget set, even when provided with a larger computational budget through extensive hyperparameter tuning. In contrast, although strong baselines such as L1-Sparse and SCRUB achieve unlearning performance comparable to MIMU, they are substantially more computationally demanding, making them less practical for time-sensitive or large-scale unlearning scenarios. MIMU offers a scalable solution that combines strong unlearning performance with low computational overhead.

4.3 Ablation study

We conduct an ablation study to analyze the impact of two key hyperparameters in MIMU: (1) the number of representative samples and (2) the top-κ masking threshold determined by the top-κ weights selected for unlearning. We evaluate these factors on both CIFAR-10 and ImageNet-100 by measuring their effect on the retain-set and forget-set accuracies. For the ablation results on MNIST and FMNIST see Appendix E.

Figure 5 presents the results on ImageNet-100. We observe that MIMU achieves effective unlearning even when using a relatively small number of representative samples and a small top-κ threshold. Simultaneously, the method drives the forget-set accuracy close to zero while largely preserving the retain-set accuracy.

As the masking threshold increases, the forget-set accuracy decreases rapidly, indicating that the weights identified by MIMU are highly influential for the target forget classes. This sharp decline suggests that the

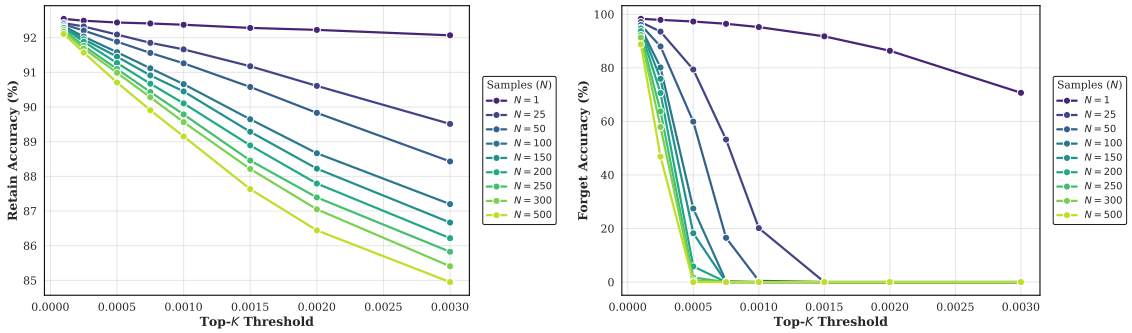


Figure 5: Ablation study of the number of representative samples and top-κ masking threshold in MIMU. Results show class-level unlearning performance on ImageNet-100, measured in terms of retain-set and forget-set accuracies after unlearning.

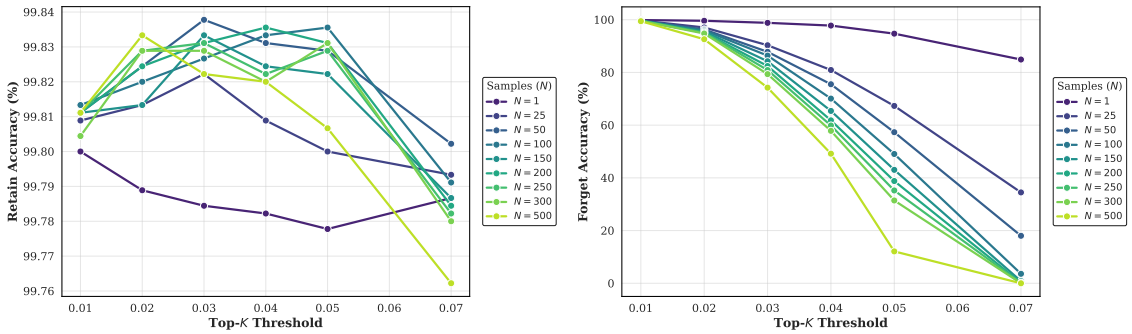


Figure 6: Ablation study of the number of representative samples and top-κ masking threshold in MIMU. Results show class-level unlearning performance on CIFAR10, measured in terms of retain-set and forget-set accuracies after unlearning.

most impactful forget-related information is concentrated within a relatively small subset of weights. The retain-set accuracy degrades at slower rate that indicates the selected weights primarily encode information associated with the forget set while contributing less to the retained classes.

The results for CIFAR-10 are shown in Figure 6. Similar to ImageNet-100, MIMU causes a substantial reduction in forget-set accuracy while maintaining the retain-set performance. However, compared to ImageNet-100, we employ larger masking thresholds.

4.4 Saliency Map Shift after Unlearning

Understanding how prediction patterns change after an unlearning operation is critical for evaluating the safety and reliability of the resulting model. In particular, even when an unlearned model preserves prediction accuracy on retain data, a shift in its internal decision-making process can raise concerns about what features the model relies on to make predictions; whether it begins to depend on superficial or spurious cues Xiao et al. (2021).

To assess this risk, we conduct a detailed prediction pattern analysis using saliency maps Simonyan et al. (2014), as shown in Figure 4. For forget set (top two rows), larger saliency difference (more red dots) is better, whereas, for retain set (bottom two rows), smaller saliency difference is better. Our results show that several strong baselines, including RL, EU, L1-Sparse, and NegGrad+, exhibit pronounced prediction pattern shifts on retain data points. This observation is consistent with our earlier findings in Table 1, where these methods demonstrate suboptimal prediction preservation on the retain set.

Interestingly, although SCRUB and MIMU achieve comparable unlearning performance in terms of accuracy, their internal behaviors differ substantially. Specifically, SCRUB induces a much stronger prediction pattern

shift on retain samples than MIMU. This result aligns well with our goal of minimally invasive unlearning, as MIMU preserves more consistent model interpretations and avoids dramatic changes in the features used for prediction. More observations are listed in Appendix D.

5 Conclusion

In this paper, we introduced MIMU, a model control–based framework for machine unlearning tasks. Unlike existing paradigms that modify model parameters, MIMU takes a different route by controlling a model's predictive behavior through a pre-trained control unit. This design leads to competitive unlearning performance while providing substantial efficiency gains. Our experiments further show that MIMU consistently suppresses the predictability of forget sets. Together, these results highlight model control as a promising new direction for sustainable machine unlearning.

Impact Statement

We propose a machine unlearning method based on posterior control that efficiently responds to near–real-time unlearning requests without incurring significant computational overhead, enabling an inference-time–optimization–free unlearning paradigm. This approach is particularly well suited for online services that require rapid intervention on deployed machine learning models to mitigate undesired behaviors caused by bad training samples while avoiding substantial performance degradation.

The field of Machine Unlearning offers many useful techniques for ensuring safer and more reliable machine learning models. These techniques can also potentially be used unethically by bad actors who wish to censor or suppress information within these models for malicious or selfish reasons.

References

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget, 2018. URL `https://arxiv.org/abs/1711.09601`.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning, 2020. URL `https://arxiv.org/abs/1912.03817`.

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)*, pp. 141–159. IEEE, 2021.

Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *2015 IEEE Symposium on Security and Privacy*, pp. 463–480, 2015. doi: 10.1109/SP.2015.35.

Sungmin Cha, Sungjun Cho, Dasol Hwang, Honglak Lee, Taesup Moon, and Moontae Lee. Learning to unlearn: Instance-wise unlearning for pre-trained classifiers, 2023.

Tianqi Chen, Emily B. Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo, 2014. URL `https://arxiv.org/abs/1402.4102`.

Vikram S Chundawat, A. Tarun, Murari Mandal, and Mohan Kankanhalli. Zero-shot machine unlearning, 2022.

Weilin Cong, Morteza Ramezani, and Mehrdad Mahdavi. On provable benefits of depth in training graph convolutional networks. *Advances in Neural Information Processing Systems*, 34:9936–9949, 2021.

A. Feder Cooper, Christopher A. Choquette-Choo, Miranda Bogen, Kevin Klyman, Matthew Jagielski, Katja Filippova, Ken Liu, Alexandra Chouldechova, Jamie Hayes, Yangsibo Huang, Eleni Triantafillou, Peter Kairouz, Nicole Elyse Mitchell, Niloofar Mireshghallah, Abigail Z. Jacobs, James Grimmelmann, Vitaly Shmatikov, Christopher De Sa, Ilia Shumailov, Andreas Terzis, Solon Barocas, Jennifer Wortman Vaughan,

danah boyd, Yejin Choi, Sanmi Koyejo, Fernando Delgado, Percy Liang, Daniel E. Ho, Pamela Samuelson, Miles Brundage, David Bau, Seth Neel, Hanna Wallach, Amy B. Cyphert, Mark Lemley, Nicolas Papernot, and Katherine Lee. Machine unlearning doesn't do what you think: Lessons for generative AI policy and research. In *The Thirty-Ninth Annual Conference on Neural Information Processing Systems Position Paper Track*, 2025. URL https://openreview.net/forum?id=mfd6GRW4Az.

Ratneel Deo, Scott Sisson, Jody M. Webster, and Rohitash Chandra. Compact bayesian neural networks via pruned mcmc sampling, 2025. URL https://arxiv.org/abs/2501.06962.

Meng Ding, Rohan Sharma, Changyou Chen, Jinhui Xu, and Kaiyi Ji. Understanding fine-tuning in approximate unlearning: A theoretical perspective, 2025. URL https://arxiv.org/abs/2410.03833.

Xin Dong, Shangyu Chen, and Sinno Jialin Pan. Learning to prune deep neural networks via layer-wise optimal brain surgeon, 2017. URL https://arxiv.org/abs/1705.07565.

Teresa Dorszewski, Lenka Tětková, Robert Jenssen, Lars Kai Hansen, and Kristoffer Knutsen Wickstrøm. From colors to classes: Emergence of concepts in vision transformers, 2025. URL https://arxiv.org/abs/2503.24071.

Chongyu Fan, Jiancheng Liu, Yihua Zhang, Eric Wong, Dennis Wei, and Sijia Liu. Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation, 2024. URL https://arxiv.org/abs/2310.12508.

Thomas Fel, Louis Bethune, Andrew Kyle Lampinen, Thomas Serre, and Katherine Hermann. Understanding visual feature reliance through the lens of complexity, 2024. URL https://arxiv.org/abs/2407.06076.

Jack Foster, Stefan Schoepf, and Alexandra Brintrup. Fast machine unlearning without retraining through selective synaptic dampening. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(11): 12043–12051, Mar. 2024. doi: 10.1609/aaai.v38i11.29092. URL https://ojs.aaai.org/index.php/AAAI/article/view/29092.

Shashwat Goel, Ameya Prabhu, Amartya Sanyal, Ser-Nam Lim, Philip Torr, and Ponnurangam Kumaraguru. Towards adversarial evaluations for inexact machine unlearning, 2023. URL https://arxiv.org/abs/2201.06640.

Vinay Chakravarthi Gogineni and E. Nadimi. Efficient knowledge deletion from trained models through layer-wise partial machine unlearning, 2024.

Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal sunshine of the spotless net: Selective forgetting in deep networks, 2020. URL https://arxiv.org/abs/1911.04933.

Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens van der Maaten. Certified data removal from machine learning models, 2023. URL https://arxiv.org/abs/1911.03030.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.

Jinghan Jia, Jiancheng Liu, Parikshit Ram, Yuguang Yao, Gaowen Liu, Yang Liu, Pranay Sharma, and Sijia Liu. Model sparsity can simplify machine unlearning, 2024. URL https://arxiv.org/abs/2304.04934.

Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. Towards unbounded machine unlearning, 2023. URL https://arxiv.org/abs/2302.09880.

Alex Labach and Shahrokh Valaee. A framework for neural network pruning using gibbs distributions. In *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, pp. 1–6. IEEE, December 2020. doi: 10.1109/globecom42002.2020.9322333. URL http://dx.doi.org/10.1109/GLOBECOM42002.2020.9322333.

Junde Li and Swaroop Ghosh. Random relabeling for efficient machine unlearning. *arXiv preprint arXiv:2305.12320*, 2023.

Nathaniel Li, Alexander Pan, Anjali Gopal, Summer Yue, Daniel Berrios, Alice Gatti, Justin D. Li, Ann-Kathrin Dombrowski, Shashwat Goel, Long Phan, Gabriel Mukobi, Nathan Helm-Burger, Rassin Lababidi, Lennart Justen, Andrew B. Liu, Michael Chen, Isabelle Barrass, Oliver Zhang, Xiaoyuan Zhu, Rishub Tamirisa, Bhrugu Bharathi, Adam Khoja, Zhenqi Zhao, Ariel Herbert-Voss, Cort B. Breuer, Samuel Marks, Oam Patel, Andy Zou, Mantas Mazeika, Zifan Wang, Palash Oswal, Weiran Lin, Adam A. Hunt, Justin Tienken-Harder, Kevin Y. Shih, Kemper Talley, John Guan, Russell Kaplan, Ian Steneker, David Campbell, Brad Jokubaitis, Alex Levinson, Jean Wang, William Qian, Kallol Krishna Karmakar, Steven Basart, Stephen Fitz, Mindy Levine, Ponnurangam Kumaraguru, Uday Tupakula, Vijay Varadharajan, Ruoyu Wang, Yan Shoshitaishvili, Jimmy Ba, Kevin M. Esvelt, Alexandr Wang, and Dan Hendrycks. The wmdp benchmark: Measuring and reducing malicious use with unlearning, 2024. URL https://arxiv.org/abs/2403.03218.

Xiang Lisa Li and Alexander M. Rush. Posterior control of blackbox generation, 2020. URL https://arxiv.org/abs/2005.04560.

Xuran Li, Jingyi Wang, Xiaohan Yuan, Peixin Zhang, Zhan Qin, Zhibo Wang, and Kui Ren. Prune: A patching based repair framework for certifiable unlearning of neural networks, 2025.

Yi-An Ma, Tianqi Chen, and Emily B. Fox. A complete recipe for stochastic gradient mcmc, 2015. URL https://arxiv.org/abs/1506.04696.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables, 2017. URL https://arxiv.org/abs/1611.00712.

Saber Malekmohammadi, H. Lee, and Li Xiong. Sharpness-aware parameter selection for machine unlearning, 2025.

Ronak R. Mehta, Sourav Pal, Vikas Singh, and Sathya Ravi. Deep unlearning via randomized conditionally independent hessians, 2022.

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

Quoc Phong Nguyen, Ryutaro Oikawa, Dinil Mon Divakaran, Mun Choon Chan, and Bryan Kian Hsiang Low. Markov chain monte carlo-based machine unlearning: Unlearning what needs to be forgotten. In *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '22, pp. 351–363. ACM, May 2022. doi: 10.1145/3488932.3517406. URL http://dx.doi.org/10.1145/3488932.3517406.

Thanh Tam Nguyen, Thanh Trung Huynh, Zhao Ren, Phi Le Nguyen, Alan Wee-Chung Liew, Hongzhi Yin, and Quoc Viet Hung Nguyen. A survey of machine unlearning. *ACM Transactions on Intelligent Systems and Technology*, 16(5):1–46, 2025.

Hojjat Salehinejad and Shahrokh Valaee. Edropout: Energy-based dropout and pruning of deep neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 33(10):5279–5292, 2022. doi: 10.1109/TNNLS.2021.3069970.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In Yoshua Bengio and Yann LeCun (eds.), *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Workshop Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6034.

Anvith Thudi, Gabriel Deza, Varun Chandrasekaran, and Nicolas Papernot. Unrolling sgd: Understanding factors influencing machine unlearning, 2022a. URL https://arxiv.org/abs/2109.13398.

Anvith Thudi, Hengrui Jia, Ilia Shumailov, and Nicolas Papernot. On the necessity of auditable algorithmic definitions for machine unlearning, 2022b. URL https://arxiv.org/abs/2110.11891.

Reihaneh Torkzadehmahani, Reza Nasirigerdeh, G. Kaissis, D. Rueckert, G. Dziugaite, and Eleni Triantafillou. Improved localized machine unlearning through the lens of memorization, 2024.

Weiqi Wang, Zhiyi Tian, Chenhan Zhang, and Shui Yu. Machine unlearning: A comprehensive survey. *arXiv preprint arXiv:2405.07406*, 2024.

Alexander Warnecke, Lukas Pirch, Christian Wressnegger, and Konrad Rieck. Machine unlearning of features and labels, 2023. URL `https://arxiv.org/abs/2108.11577`.

Leon Wichert and Sandipan Sikdar. Rethinking evaluation methods for machine unlearning. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 4727–4739, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.271. URL `https://aclanthology.org/2024.findings-emnlp.271/`.

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*, volume 36, pp. 8675–8682, 2022.

Kai Yuanqing Xiao, Logan Engstrom, Andrew Ilyas, and Aleksander Madry. Noise or signal: The role of image backgrounds in object recognition. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=gl3D-xY7wLq`.

Sang Michael Xie and Stefano Ermon. Reparameterizable subset sampling via continuous relaxations, 2021. URL `https://arxiv.org/abs/1901.10517`.

Yangyan Xu, Cong Cao, Fangfang Yuan, Rongxin Mi, Nannan Sun, Dakui Wang, and Yanbing Liu. Efficient one-shot pruning of large language models with low-rank approximation. In *2024 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 1036–1041, 2024. doi: 10.1109/SMC54092.2024.10831406.

Haonan Yan, Xiaoguang Li, Ziyao Guo, Hui Li, Fenghua Li, and Xiaodong Lin. Arcane: An efficient architecture for exact machine unlearning. In Lud De Raedt (ed.), *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pp. 4006–4013. International Joint Conferences on Artificial Intelligence Organization, 7 2022a. doi: 10.24963/ijcai.2022/556. URL `https://doi.org/10.24963/ijcai.2022/556`. Main Track.

Haonan Yan, Xiaoguang Li, Ziyao Guo, Hui Li, Fenghua Li, and Xiaodong Lin. Arcane: An efficient architecture for exact machine unlearning. In *IJCAI*, volume 6, pp. 19, 2022b.

Table 4: Summary of datasets used in this paper. For each dataset, the corresponding model architecture is specified. The internal dimensions of the MLP models are listed in the table, while for ResNet we follow the original architecture with one additional hidden layer inserted before the classification layer, whose dimension is also reported. For ImageNet-100, the model was initialized with weights trained on ImageNet.

CIFAR-10	Benchmark	Image	60,000	$32 \times 32 \times 3$	10	No	Yes	ResNet18 (512, 128)
MNIST	Benchmark	Image	60,000	$28 \times 28 \times 1$	10	No	Yes	MLP (128, 64)
Fashion-MNIST	Benchmark	Image	70,000	$28 \times 28 \times 1$	10	No	Yes	CNN (128, 64)
ImageNet-100	Benchmark	Image	130,000	$224 \times 224 \times 3$	100	No	Yes	ResNet50(2048, 256)

A Additional Derivations

Here, we provide additional derivations of the equations in the main paper.All notations, datasets and experimental settings follow the definition of the main paper unless stated otherwise.

A.1 Equation 3

Following the Bayes rules', the posterior distribution of masking m_i can be derived as product of prior and likelihood terms

$$P(\mathbf{m}_i|(x_i, y_i), \theta) = \frac{P_\theta(x_i, y_i|\mathbf{m}_i)\, P(\mathbf{m}_i|\theta)}{P_\theta(x_i, y_i)}$$

We can re-write the likelihood term with explicit notion where we apply mask m_i to the model parameter θ

$$P_\theta(x_i, y_i|\mathbf{m}_i) \coloneqq P_{\mathbf{m}_i \odot \theta}(x_i, y_i)$$

By the properties of joint probability:

$$P_{\mathbf{m}_i \odot \theta}(x_i, y_i) = P_{\mathbf{m}_i \odot \theta}(y_i|x_i)\, P(x_i)$$

$$P_\theta(x_i, y_i) = P_\theta(y_i|x_i)\, P(x_i)$$

Thus, we have:

$$P(\mathbf{m}_i|(x_i, y_i), \theta) = \frac{P_{\mathbf{m}_i \odot \theta}(y_i|x_i)\, P(x_i)\, P(\mathbf{m}_i|\theta)}{P_\theta(y_i|x_i)\, P(x_i)}$$

which corresponds to Equation (3) in the main paper.

A.2 Equation 4

Starting with the motivating objective in Equation 2, we have

$$\begin{aligned} &\text{argmin}_\phi KL\left[Q_\phi(\mathbf{m}_i|(x_i, y_i), \theta)||P(\mathbf{m}_i|(x_i, y_i), \theta)\right] \\ &\quad s.t. \quad ||\mathbf{m}_i||_0 \leq \delta, \end{aligned} \tag{12}$$

which, by extend, is defined as following

$$\text{KL}(Q\|P) = E_{Q_\phi}\left[\log \frac{Q_\phi(\mathbf{m}_i \mid (x_i, y_i), \theta)}{P(\mathbf{m}_i \mid (x_i, y_i), \theta)}\right] \tag{13}$$

Now, considering the outcome of Equation 3 , we can further derive it to

$$\text{KL}(Q\|P) \tag{14}$$

$$= E_{Q_\phi}\left[\log \frac{Q_\phi(\mathbf{m}_i \mid (x_i, y_i), \theta)}{P_{\mathbf{m}_i \odot \theta}(y_i \mid x_i)\, P(\mathbf{m}_i \mid \theta)/P_\theta(y_i \mid x_i)}\right] \tag{15}$$

$$= -E_{Q_\phi}\left[\log P_{\mathbf{m}_i \odot \theta}(y_i \mid x_i)\right] + E_{Q_\phi}\left[P_\theta(y_i \mid x_i)\right] \tag{16}$$

$$+ \text{KL}\left(Q_\phi(\mathbf{m}_i \mid (x_i, y_i), \theta) \,\|\, P(\mathbf{m}_i \mid \theta)\right) \tag{17}$$

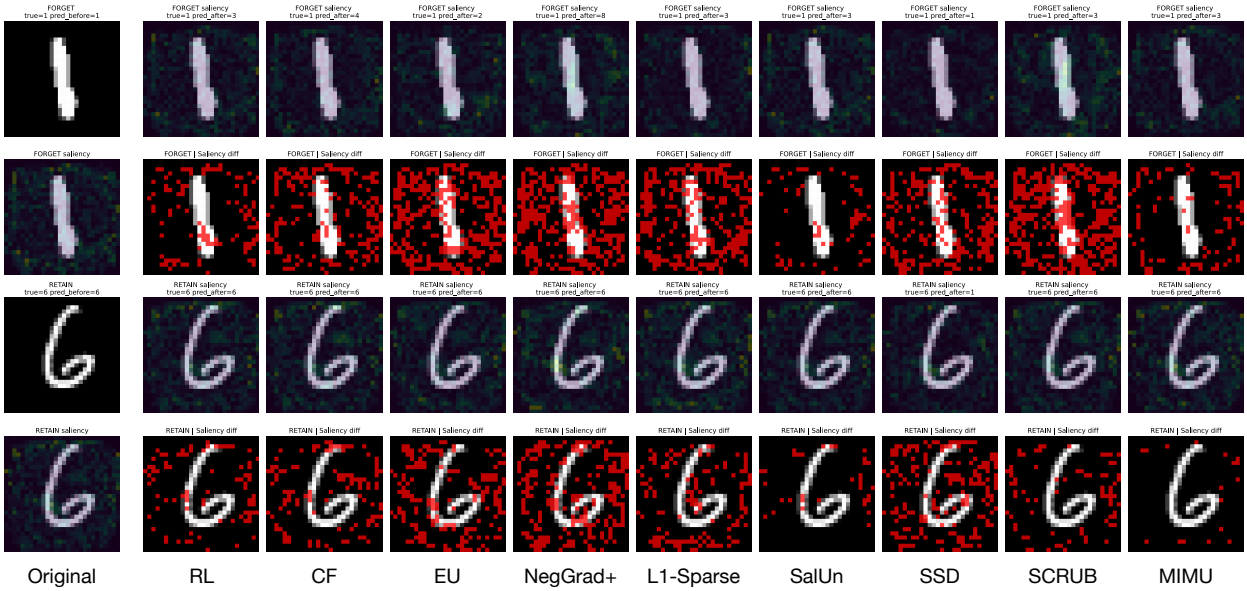


Figure 7: Extra saliency map analysis before and after unlearning on MNIST dataset. The top row shows the saliency map of a forget sample after unlearning. The second row shows the saliency difference between before and after unlearning for the forget sample. The bottom two rows show the corresponding results for a retain sample. Red dots highlight regions with notable saliency differences, which implies prediction pattern shift of target model on the data.

Note, the term $\log P_\theta(y_i \mid x_i)$ is constant as it doesn't rely on m_i. Finally, to enforce the sparsity constraint $\|m_i\|_0 \leq \delta$, we incorporate a Lagrange multiplier term, leading to the complete objective:

$$\begin{aligned} &\text{argmin}_\phi \max_\lambda -E_{Q_\phi}[\log P_{\mathbf{m}_i \odot \theta}(y_i|x_i)] + \\ &KL[Q_\phi(\mathbf{m}_i|(x_i, y_i), \theta)||P(\mathbf{m}_i|\theta)] + \lambda(||\mathbf{m}_i||_0 - \delta), \end{aligned} \tag{18}$$

B Benchmark Datasets and Model Architecture

We evaluate our method on a diverse set of benchmark datasets that span different data modalities, input dimensionalities, and model complexities, as summarized in Table 4. Specifically, we include MNIST and Fashion-MNIST as canonical grayscale image classification benchmarks with relatively low-dimensional inputs, paired with lightweight MLP and CNN architectures to reflect common practice. CIFAR-10 serves as a standard natural image benchmark with higher visual complexity, for which we adopt a ResNet18 backbone following the original design, augmented with an additional hidden layer before the classification head as detailed in Table 4. To further assess scalability and behavior in large-scale visual settings, we include ImageNet-100, a widely used subset of ImageNet, and train a ResNet50 model to capture deeper hierarchical representations.

C Hyperparameters

In Table 5, we provide detailed hyperparameter settings for the vision model, GCN training, baselines methods and MIMU masking. For baselines methods, the parameters reported in their original papers were used, and were tuned if not performing correctly.

C.1 Limitations

This work focuses on a control-based machine unlearning (MU) framework that achieves minimally invasive unlearning via learned masking, without inference-time optimization or weight updates. In the current implementation, the controller module is instantiated as a graph convolutional network (GCN), which efficiently

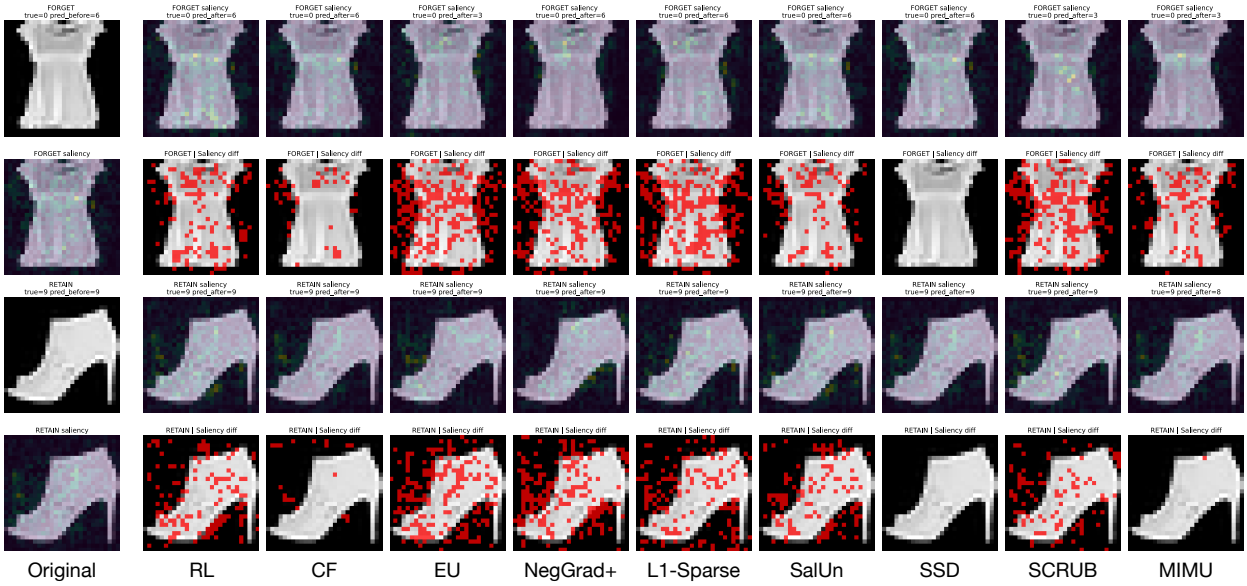


Figure 8: Extra saliency map analysis before and after unlearning on FMNIST dataset. The top row shows the saliency map of a forget sample after unlearning. The second row shows the saliency difference between before and after unlearning for the forget sample. The bottom two rows show the corresponding results for a retain sample. Red dots highlight regions with notable saliency differences, which implies prediction pattern shift of target model on the data.

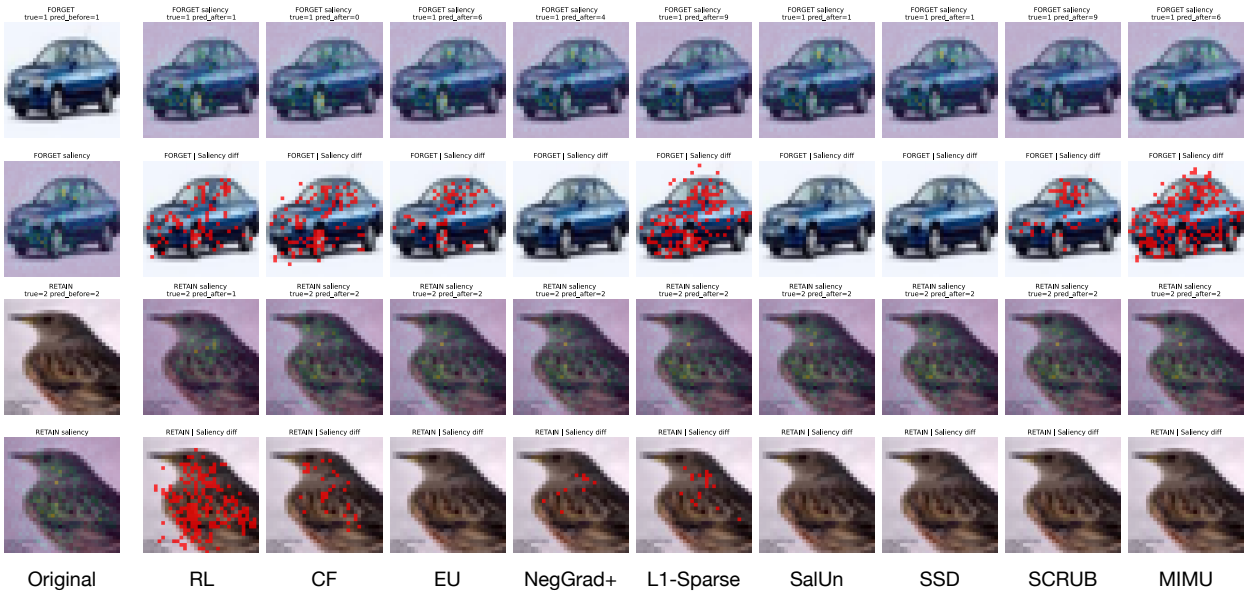


Figure 9: Extra saliency map analysis before and after unlearning on CIFAR10 dataset. The top row shows the saliency map of a forget sample after unlearning. The second row shows the saliency difference between before and after unlearning for the forget sample. The bottom two rows show the corresponding results for a retain sample. Red dots highlight regions with notable saliency differences, which implies prediction pattern shift of target model on the data.

captures parameter interactions and remains practical as model size scales. However, GCNs are known to be sensitive to training configurations Cong et al. (2021), particularly architectural choices such as network depth and the number of message-passing steps, which can pose practical challenges for less experienced practitioners. Importantly, this limitation is not intrinsic to the proposed MIMU framework. The controller is modular and can be replaced with alternative neural architectures beyond GCNs. Investigating other con-

Figure 10: Extra saliency map analysis before and after unlearning on ImageNet100 dataset. The top row shows the saliency map of a forget sample after unlearning. The second row shows the saliency difference between before and after unlearning for the forget sample. The bottom two rows show the corresponding results for a retain sample. Red dots highlight regions with notable saliency differences, which implies prediction pattern shift of target model on the data.

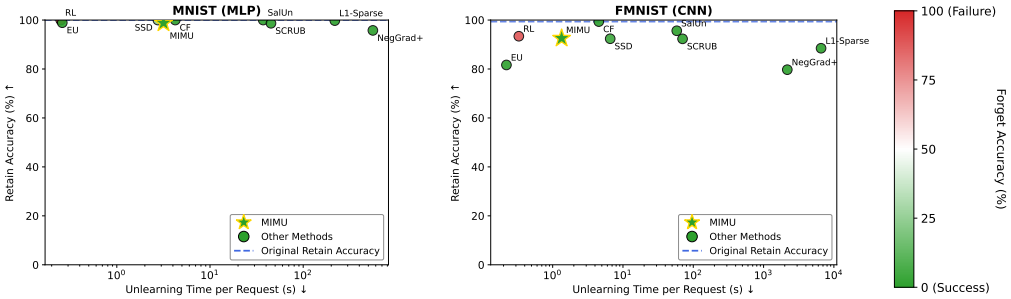


Figure 11: For every unlearning request, MIMU delivers the competitive results quickly. Other common MU techniques deliver similar performance but can cost significantly more in compute time. The graph above shows the Retain Accuracy in percentage on the Y axis, the Unlearning Time measured in wall-clock seconds on the X axis and the Forget Accuracy is measure from the hue of the marker, from red (failure - up to 100% retention) to green (success - down to 0% retention). Time values can be found in Table 6.

troller designs (e.g., transformer- or diffusion-based models) to reduce manual calibration while preserving optimization-free unlearning is a promising direction for future work.

D Additional Experimental Plots and Tables

Figures 7, 8, 9 and 10 present additional saliency map analysis results on the MNIST,CIFAR-10 and ImageNet-100 datasets. Consistent with the main paper, most existing approximate MU methods induce significant prediction pattern changes after unlearning, whereas MIMU and SalUn generally preserve the model's original prediction behavior and cause less disruption. Such the significancy of such observation varies based on the dataset: On MNIST and FMNIST datasets, the corruption of prediction pattern is more clear than that of the CIFAR-10 and ImageNet datasets.

Table 5: Comprehensive Hyperparameter Configurations for Unlearning Experiments. A hyphen means the default parameter was used for class regime, or the corresponding class parameter was used for subclass regime.

Category	**Hyperparameter**	**Default**	**MNIST (MLP)**		**FMNIST (CNN)**		**CIFAR10 (ResNet18)**		**ImageNet100 (ResNet50)**	
			Class	Subclass	Class	Subclass	Class	Subclass	Class	Subclass
Vision Training	Epochs	100	-	-	40	-	200	-	20	-
	Learning Rate	3×10^{-4}	-	-	10^{-3}	-	0.1	-	0.01	-
	Batch Size	256	-	-	-	-	-	-	128	-
	Weight Decay	0.01	-	-	-	-	5×10^{-4}	-	10^{-4}	-
	Optimizer	AdamW	-	-	-	-	SGD	-	SGD	-
	Opt. Momentum	0.9	-	-	-	-	-	-	-	-
	LR Scheduler	None	-	-	-	-	Cosine	-	Cosine	-
GCN Training	Epochs	11	-	-	12	-	8	-	8	-
	Learning Rate	10^{-3}	-	-	-	-	-	-	-	-
	mask_K (Utility)	0.1	-	-	0.05	-	-	-	0.01	-
	Sample Rate	1.0	0.05	-	0.05	-	0.05	-	0.01	-
	Hidden Dim	32	-	-	-	-	-	-	-	-
	Gumbel Temp	0.1	-	-	-	-	-	-	-	-
CF	Steps	50	2000	-	2000	-	500	-	10000	-
	Learning Rate	10^{-3}	0.02	-	0.02	-	0.02	-	0.002	-
EU	Steps	50	1000	-	1000	-	-	-	10000	-
	Learning Rate	10^{-3}	-	-	-	-	-	-	-	-
RL	Steps	50	1000	-	1000	-	-	-	1000	-
	Learning Rate	10^{-3}	-	-	-	-	-	-	-	-
NegGrad+	Alpha (α)	0.9	0.99	-	0.99	-	0.99	-	0.99	-
	Steps	10	200	-	200	-	20	-	-	-
	Weight Decay	0.1	-	-	-	-	-	-	-	-
L1-Sparse	Alpha (α)	0.2	0.01	-	0.005	-	0.005	-	0.005	-
	Steps	50	100	-	100	-	-	-	-	-
	Warmup	5	-	-	-	-	-	-	-	-
SalUn	Threshold	0.15	-	-	0.2	-	0.5	-	-	-
	Momentum	1.0	0.9	-	0.9	-	-	-	0.9	-
SSD	Dampening (λ)	1.0	-	-	-	-	-	-	-	-
	Selection (α)	10.0	2.0	2.0	20.0	20.0	5.0	5.0	-	-
SCRUB	SGDA Epochs	10	-	-	-	-	-	-	-	-
	Learning Rate	5×10^{-4}	-	-	-	-	-	-	-	-
	M-Steps	3	-	-	-	-	-	-	-	-
	Gamma (γ)	1.0	-	-	-	-	-	-	-	-
	Alpha (α)	0.5	-	-	-	-	-	-	-	-
	Beta (β)	0.0	-	-	-	-	-	-	-	-
	KD Temp (T)	2	-	-	-	-	-	-	-	-
MIMU Eval	topK (Pruning)	0.1	0.05	-	0.0025	0.002	0.03	0.002	10^{-3}	2×10^{-4}
	Representatives	100	300	-	600	-	-	-	-	-
	Batch Size	32	-	-	-	-	-	-	-	-

Table 6: Unlearning time per request (wall-clock, in seconds). Lower values represent faster unlearning requests. Results are averaged over 5 independent runs. MIMU clearly demonstrates rapid unlearning across across architectures.

Dataset	RL	CF	EU	NegGrad+	L1-Sparse	SalUn	SSD	SCRUB	MIMU
MNIST (MLP)	0.25 ± 0.01	4.24 ± 0.39	0.26 ± 0.01	559.20 ± 2.68	218.40 ± 3.29	37.24 ± 3.98	2.76 ± 0.10	45.12 ± 0.26	3.17 ± 0.21
FMNIST (CNN)	0.33 ± 0.15	4.53 ± 0.07	0.22 ± 0.01	2174.17 ± 53.61	6572.42 ± 58.42	58.24 ± 0.91	6.58 ± 0.00	70.43 ± 0.90	1.34 ± 0.20
CIFAR10 (ResNet 18)	1.69 ± 0.77	5.88 ± 1.58	1.34 ± 0.25	282.43 ± 106.47	406.42 ± 102.06	140.75 ± 0.24	67.62 ± 4.65	220.54 ± 79.43	2.25 ± 0.25
ImageNet100 (ResNet 50)	2.47 ± 0.53	141.82 ± 24.61	148.93 ± 31.92	2035.78 ± 722.72	7017.55 ± 1167.38	1948.37 ± 0.78	356.86 ± 31.98	3394.54 ± 908.64	9.18 ± 2.08

Figure 11 provides additional analysis of the unlearning effectiveness and computational efficiency on MNIST and FMNIST, complementing the results presented in the main paper. The x-axis represents the average wall-clock unlearning time required to process a single request, while the y-axis reports the retain accuracy after unlearning. The color gradient indicates the forget accuracy, where green shades correspond to lower forget accuracy and therefore more effective forgetting. The dashed horizontal line denotes the accuracy of the original model prior to unlearning. MIMU is highlighted using star markers, whereas competing methods are represented by circular markers.

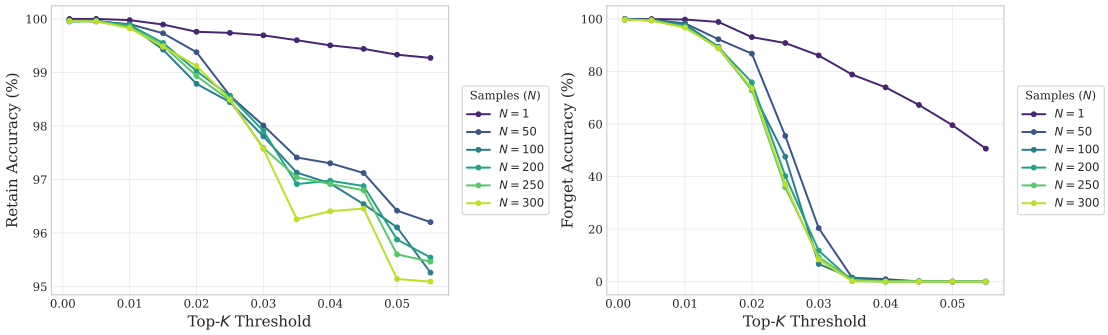


Figure 12: Ablation study of the number of representative samples and top-κ masking threshold in MIMU. Results show class-level unlearning performance on MNIST, measured in terms of retain-set and forget-set accuracies after unlearning.

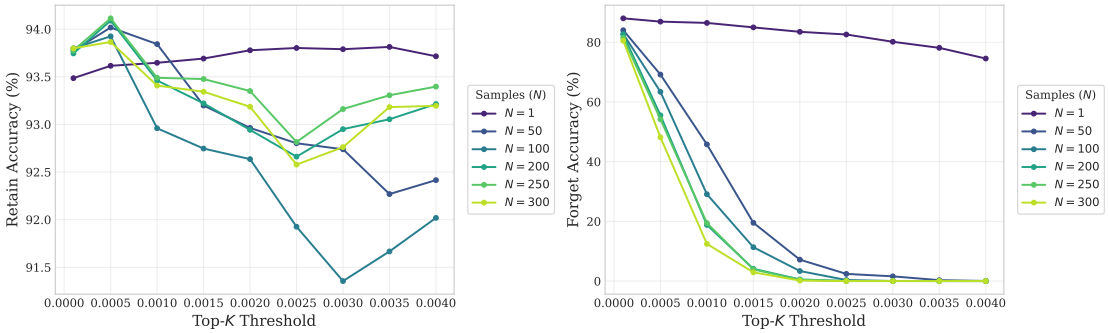


Figure 13: Ablation study of the number of representative samples and top-κ masking threshold in MIMU. Results show class-level unlearning performance on FMNIST, measured in terms of retain-set and forget-set accuracies after unlearning.

Table 6 reports the average wall-clock execution time (in seconds) required by each machine unlearning method, measured over five independent runs. The experiments tested MIMU and other unlearning methods on the MLP (MNIST), CNN (FMNIST), ResNet-18 (CIFAR-10), and ResNet-50 (ImageNet-100).

E Ablation study

We conduct an ablation study to analyze the impact of two key hyperparameters in MIMU: (1) the number of representative samples and (2) the top-κ masking threshold determined by the top-κ weights selected for unlearning.

Figure 12 presents the results on MNIST. We observe that MIMU achieves effective unlearning even when using a relatively small number of representative samples and a small top-κ threshold. Simultaneously, the method drives the forget-set accuracy close to zero while preserving the retain-set accuracy. MNIST sees the most deterioration of the retain accuracy with increased top-κ thresholds.

The results for FMNIST are shown in Figure 13. Similar to CIFAR-10, MIMU causes a substantial reduction in forget-set accuracy while maintaining the retain-set performance. Similar to ImageNet-100, FMNIST requires very small top-κ thresholds to achieve competitive unlearning performance.

F Broader Vision: Interpretability Through MIMU

Our framework has broader impact beyond efficient and sustainable machine unlearning, as the MIMU masking mechanism also offers a principled route toward explaining model predictions. By identifying a

minimal subset of parameters whose masking reliably removes the influence of a forget set, MIMU implicitly characterizes which internal components are most responsible for specific behaviours, in line with masking-based explanation methods that attribute predictions to a sparse subset of parameters, inputs, or circuits. This dual use of masks for both unlearning and explanation can strengthen the evidence that minimally invasive interventions on a small, well-identified subnetwork are sufficient to control model behaviour, rather than requiring opaque, global parameter changes. In practice, this connection can support more transparent auditing pipelines in which the same machinery used to honour data-deletion requests (e.g., for regulatory compliance or user rights) also yields interpretable summaries of how and where sensitive information was encoded. More broadly, tying unlearning to sparse, explanation-driven interventions encourages the design of models whose internal structure is explicitly aligned with controllability and accountability, helping to bridge the gap between privacy-preserving ML and explainable AI.