
White Rose Research Online

eprints@whiterose.ac.uk
https://eprints.whiterose.ac.uk
Universities of Leeds, Sheffield and York

Deposited via The University of Sheffield.

White Rose Research Online URL for this paper:
https://eprints.whiterose.ac.uk/id/eprint/247742/

Version: Preprint

Preprint:
Kumaran, M.C., Unni, M.P., Kouni, V. et al. (Submitted: 2026) Similarity-aware machine unlearning. [Preprint - arXiv] (Submitted)

https://doi.org/10.48550/arXiv.2608.00246

© 2026 The Author(s). This preprint is made available under a Creative Commons Attribution 4.0 International License. (https://creativecommons.org/licenses/by/4.0/)

Reuse
This article is distributed under the terms of the Creative Commons Attribution (CC BY) licence. This licence allows you to distribute, remix, tweak, and build upon the work, even commercially, as long as you credit the authors for the original work. More information and the full terms of the licence here: https://creativecommons.org/licenses/

Takedown
If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.





Similarity-Aware Machine Unlearning

Madhavan Citalamangalam Kumaran[1][0009−0008−9954−090X], Midhun Parakkal Unni[2,3][0000−0002−2913−8869], Vicky Kouni[4][0000−0002−1316−8967], and Haripriya Harikumar[1][0000−0001−9918−381X]

[1] Department of Computer Science, University of Manchester, UK
[2] Centre for Machine Intelligence, University of Sheffield, UK
[3] School of Computer Science, University of Sheffield, Sheffield, UK
[4] Paris Dauphine - PSL University, Paris, France
madhavan.citalamangalamkumaran@student.manchester.ac.uk, m.parakkalunni@sheffield.ac.uk, vasiliki.kouni@lamsade.dauphine.fr, haripriya.harikumar@manchester.ac.uk

arXiv:2608.00246v1 [cs.LG] 31 Jul 2026

Abstract. Machine unlearning removes the influence of user-specified training examples from a trained model, avoiding the need to retrain it from scratch. Localization-based methods improve unlearning efficiency by identifying a subset of influential model parameters. However, existing approaches select parameters based solely on forget-set importance, neglecting their role in retained dataset and often causing collateral damage to semantically similar retained examples. We address this limitation with a retain-aware localization method that considers parameter importance to both forgotten and retained data. We also introduce a retain-similar evaluation set, constructed using cosine similarity in the model embedding space, to directly measure collateral damage. Across eleven experimental settings on CIFAR-10 dataset and ResNet18 model, our method consistently reduces collateral damage while improving standard unlearning metrics, demonstrating the effectiveness of retain-aware localization for similarity-aware machine unlearning.

Keywords: Similarity · Unlearning · collateral damage · embedding space · localization

1 Introduction

Machine learning models can memorize information about individual training examples to varying degrees [26], creating privacy risks even when the underlying training records are not directly accessible. Membership-inference attacks can reveal whether an example was used for training [23], model-inversion attacks [11] can expose sensitive attributes [21], and generative models can reproduce passages from their training corpora verbatim [6]. These risks are amplified when models are trained on personal or sensitive data [20, 21]. Data-protection frameworks such as the General Data Protection Regulation [25] and California Consumer Privacy Act [19] establish rights concerning the erasure or deletion of personal data. Together, these privacy risks and data-deletion rights motivate mechanisms for removing the residual influence from models.

Machine unlearning [5, 2, 27] addresses this problem by modifying a trained model so that its behavior approximates that of an oracle model retrained without a user-specified subset of training examples, referred to as the forget set. Retraining from scratch on the remaining data provides the best solution, but can be computationally expensive, particularly when deletion requests are frequent. Approximate unlearning [17] methods therefore seek to approximate the behavior of the retrained model at substantially lower computational cost. Localized unlearning [24] methods reduce this cost by restricting model modifications to a selected subset of parameters that are influential for the forget set.

However, importance to the forget set does not imply specificity to the forget set. A parameter may be highly influential for forgotten examples while also supporting retained examples. This overlap is especially consequential when forgotten and retained examples occupy nearby regions of the learned representation space and may depend on shared features, as shown in Fig. 1 (visually similar images in red and green boxes naturally cluster closely in their embedding space). Consequently, forget-only localization can modify parameters used by both sets, causing unintended degradation on representation-similar retained examples. We refer to this representational overlap as *feature entanglement* and to the resulting degradation as *collateral damage*. Because such damage may be concentrated among a small subset of retained examples, aggregate test and retain-set accuracy may fail to reveal it.

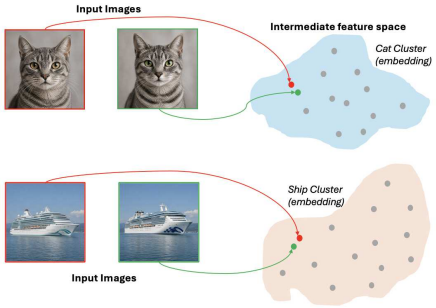


Fig. 1. Intermediate feature-space visualization showing that semantically similar samples are embedded close to one another within class-specific clusters. The image to forget (red) and the image to retain (green) occupy very close neighboring regions of the feature manifold, indicating shared feature representations.

To address this limitation, we propose a retain-aware localization framework. We compute criticality scores separately on the forget set and the retain set, and combine them to prioritize neurons that are important to the forget set but comparatively less important to the retain set. To measure collateral damage directly, we also construct a retain-similar evaluation set. For each forgotten example, this set contains its nearest same-class retained neighbor according to cosine similarity in the original model's embedding space. Our contributions are as follows.

– We identify collateral damage to semantically similar retained examples as a key limitation of existing localization-based machine unlearning methods and relate this phenomenon to shared learned representations.

- We propose a retain-aware localization framework that incorporates parameter importance with respect to both forgotten and retained data, enabling similarity-aware unlearning.
- We introduce a retain-similar evaluation protocol and demonstrate across eleven experimental settings that our approach consistently reduces collateral damage while improving standard unlearning metrics.

2 Background

We introduce the notation and definitions used throughout the paper before presenting our method. Let $\mathcal{D} := \bigcup_{n\geq 1}(\mathcal{X} \times \mathcal{Y})^n$ denote the set of all datasets and $\mathcal{A}$ a randomized training algorithm,$\mathcal{A} : \mathcal{D} \to \Delta(\mathcal{H})$, where $\Delta(\mathcal{H})$ denotes the set of all probability distributions over the hypothesis space $\mathcal{H}$. The hypothesis space $\mathcal{H} := \{f_\theta : \mathcal{X} \to \mathcal{Y} \mid \theta \in \mathbb{R}^d\}$ is the set of all models that the learning algorithm $\mathcal{A}$ produces, parameterized by weight vectors $\theta \in \mathbb{R}^d$. We denote by $f_{\theta^o} \sim \mathcal{A}(D_{train} \subset \mathcal{D})$ the original model parameterised by θ^o when trained on the training set $D_{train} \subset \mathcal{D}$ prior to unlearning.

Let the forget set be denoted by $D_f \subset D_{train}$, consisting of the data requested for removal. The retain set is defined as,$D_r = D_{train} \setminus D_f$ representing the remaining training data. We denote $f_{\theta^r} \sim \mathcal{A}(D_r)$ as oracle model and θ^r as the associated parameters by training from scratch only on the retain set alone. Machine unlearning is defined as the process of modifying the parameters of a model from $\mathcal{A}(D_{train})$, in order to remove the influence of D_f without compromising performance on the retain set D_r. Formally,

Definition 1 (Unlearning [3]). *An unlearning algorithm $\mathcal{U}$ is said to unlearn a forget set $D_f \subset D_{train}$ if the unlearned model $f_{\theta^u} \sim \mathcal{U}(f_{\theta^o}, D_f, D_r)$ is indistinguishable from the retrained model $f_{\theta^r} \sim \mathcal{A}(D_r)$, where $D_r = D_{train} \setminus D_f$. Formally, the output distributions of f_{θ^u} and f_{θ^r} are identical on the evaluation distribution.*

Definition 1 characterizes the goal of machine unlearning where the unlearned model f_{θ^u} should behave indistinguishably from a model retrained from scratch on the retain set D_r. One way to satisfy this definition is exact unlearning, where the model is retrained from scratch using only D_r, to produce $f_{\theta^r} \sim \mathcal{A}(D_r)$. Although exact unlearning directly satisfies the definition, retraining with large models and datasets can be computationally expensive. Therefore, practical approaches often employ approximate unlearning, where an unlearning algorithm $\mathcal{U}$ modifies the original model f_{θ^o} to obtain $f_{\theta^u} \sim \mathcal{U}(f_{\theta^o}, D_f, D_r)$. The objective is to have an unlearned model f_{θ^u} that produces outputs closely matched to those of the retrained model f_{θ^r}, while requiring significantly less computation. More precisely, machine unlearning approach aims for the following:

$$t\big(\mathcal{U}(f_{\theta^o}, D_f, D_r)\big) \ll t\big(\mathcal{A}(D_r)\big), \tag{1}$$

$$\mathcal{U}(f_{\theta^o}, D_f, D_r) \approx \mathcal{A}(D_r), \tag{2}$$

where $t(\cdot)$ denotes the execution time of the unlearning or training process. We use the following definition to operationalise evaluation of unlearning in line with Torkzadehmahani et al. [24].

Definition 2 (Localized Machine Unlearning [24]). *Let $\theta^o = \{\theta_k\}_{k=1}^N$ denote the parameters of the original model, where N is the total number of trainable parameters. An unlearning algorithm $\mathcal{U}$ is said to perform localized machine unlearning if, given a forget set D_f, it identifies and modifies only a subset of parameters $\theta_s \subset \theta^o, |\theta_s| = n < N$, while leaving the remaining parameters unchanged during unlearning. The resulting unlearned model is from $\mathcal{U}(f_{\theta^o}, D_f, D_r)$, where updates are restricted to the selected parameter subset θ_s.*

Definition 3 (Memorization Score [26]). *Memorization score for a data point x_i with label y_i in a training set D_{train}, $(x_i, y_i) \in D_{train}$ with a randomized training algorithm $\mathcal{A}$ is given as follows,*

$$mem(\mathcal{A}, D, x_i) = \Pr_{f_\theta \sim \mathcal{A}(D_{train})}(f_\theta(x_i) = y_i) - \Pr_{f_{\theta'} \sim \mathcal{A}(D_{train} \setminus (x_i, y_i))}(f_{\theta'}(x_i) = y_i),$$

where f_θ is a model parameterized by θ sampled from the distribution $\mathcal{A}(D_{train})$, and $f_{\theta'}$ is a model parameterized by θ' sampled from the distribution $\mathcal{A}(D_{train}) \setminus (x_i, y_i)$. The first term considers model parameters trained on the entire dataset, while the second is the model parameters trained without the example (x_i, y_i). A high memorization score indicates that excluding the example causes a significant change in the performance of the model for that example.

3 Proposed Similarity Aware Localized Unlearning

We propose a Similarity-Aware Localized Unlearning framework that extends the Deletion by Example Localization (DEL) [24], by explicitly incorporating intermediate feature-space similarity between forget and retain set images into the unlearning process. Existing localization-based unlearning methods [24] identify influential layers and neurons primarily from the forget set (D_f), without considering whether these parameters also contribute to semantically similar samples in the retain set (D_r). As a result, modifying localized parameters may affect learned representation from the retain set.

Our key observation is that semantically similar forget and retain samples are embedded in nearby regions of the intermediate feature space (in Fig. 1) and therefore tend to activate overlapping subsets of neurons. Motivated by this observation, we perform localization using information from both the forget and retain sets. To address this limitation, our method uses forget samples and constrains the unlearning process with the retain set to preserve their representations. By jointly considering parameter localization and sample similarity, we selectively modify forget-specific knowledge while minimizing collateral damage to the D_r. By jointly analyzing neuron importance across D_f and D_r, our proposed method identifies parameters that are strongly associated with the forget set while accounting for their contribution to similar retain samples. This

similarity-aware localization enables targeted removal of forget-specific knowledge while reducing collateral damage to retained knowledge and preserving overall model performance.

Definition 4 (Semantic Similarity). *Let $\phi(\cdot)$ denote the embedding representation produced by a model. For two samples x_i and x_j, the semantic similarity between their learned representations is defined using cosine similarity as*

$$\mathrm{sim}(x_i, x_j) = \frac{\phi(x_i)^\top \phi(x_j)}{\|\phi(x_i)\| \, \|\phi(x_j)\|}. \tag{3}$$

Two examples are considered semantically similar when their embedding representations exhibit high cosine similarity in the learned feature space. Consider a forget-set example $(x_i, y_i) \in D_f$ and a retain-set example $(x_j, y_j) \in D_r$, such that $\mathrm{sim}(x_i, x_j)$ is high. Since both examples occupy nearby regions in the learned representation space, they are likely to share intermediate feature representations and depend on overlapping subsets of model parameters.

Definition 5 (Similarity-Aware Localized Unlearning). *Consider $\theta^o = \{\theta_k\}_{k=1}^N$ denote the parameters of the original model f_{θ^o} and $mem(\theta_k, x_i)$ quantify the contribution of parameter θ_k to the memorization of example x_i. An unlearning algorithm $\mathcal{U}$ is said to perform similarity-aware localized unlearning if it selects a subset of parameters $\theta_s \subset \theta^o, |\theta_s| = n < N$, such that the selected parameters have high memorization contribution to the forget-set data while minimizing disruption to semantically similar retain-set data. Formally, the selected subset should satisfy the following,*

$$mem(\theta_s, x_i) \gg mem(\theta_s, x_j), \tag{4}$$

for semantically similar pairs (x_i, x_j) where $x_i \in D_f$ and $x_j \in D_r$.

3.1 Criticality scores

Definition 6 ((Parameter) criticality score). *The (parameter) criticality score s_j of the j^{th} parameter θ^o denoted as θ^o_j, where $j \in \{1, \ldots, N\}$ and N is the total number of parameters, is defined as*

$$s_j = \left| \theta^o_j \cdot g(\theta^o_j, D) \right|. \tag{5}$$

The (parameter) criticality score measures the importance of the parameter θ^o_j with respect to a given dataset D. Let $g(\theta^o_j, D)$ denote the gradient of the loss with respect to θ^o_j computed over the forget set. This identifies parameters whose current values are both large and strongly implicated in the model's predictions on the dataset. Each layer of the model consists of a set of M neurons. We aim to identify the neurons with the highest (parameter) criticality scores; to that end, we introduce the following definition.

Definition 7 (Neuron criticality score). *Let $\tilde{s}_i$ denote the sequence of (parameter) criticality scores corresponding to the parameters associated with neuron o_i, sorted in descending order. We compute the neuron criticality score co_i for neuron o_i, by averaging the top-h parameter criticality scores:*

$$co_i = \frac{1}{h} \sum_{j=1}^{h} \tilde{s}_i[j]. \tag{6}$$

Without loss of generality, in the remainder of this work we will refer to the (parameter) criticality scores simply as criticality scores.

3.2 Criticality score of similarity aware localized unlearning

We compute two sets of criticality scores for our localized unlearning framework. The first set is computed on the forget set D_f, denoted as s^f, and the second set is computed on the retain set D_r denoted as s^r. The neuron criticality scores are then used to identify the neurons that should be finetuned for unlearning. We propose three methods of localization strategies by considering the forget set and retain set: (1) the `difference` method, (2) the `weighted_difference` method, and (3) the `ratio` method. The methods are outlined below.

Difference method In the `difference` method, we used the difference between the forget set and the retain set criticality score of a neuron to identify its relevance and later for unlearning. So, the criticality score is:

$$\tilde{s}_j = \max\Big(s_j^f - s_j^r,\ 0\Big), \tag{7}$$

where s_j^f and s_j^r are the forget-set and retain-set parameter criticality scores, respectively. The clamp at zero ensures that parameters more important for the retain set than the forget set receive a score of zero and are excluded from the localization mask entirely.

Weighted Difference method In the `weighted_difference` method, we introduce a scalar weight w on the forget set criticality score as follows:

$$\tilde{s}_j = \max\Big(w \cdot s_j^f - s_j^r,\ 0\Big). \tag{8}$$

The scalar weight w allows the balance between forget-set specificity and retain-set protection to be tuned continuously. As w increases, the scoring criterion converges toward the forget-only baseline; as w approaches 1, the scoring criterion converges to the unweighted difference.

Ratio method In the `ratio` method, we propose a ratio between the criticality score of the forget set and retain set:

$$\tilde{s}_j = \frac{s_j^f}{s_j^r + \varepsilon}, \tag{9}$$

where $\varepsilon = 10^{-8}$ ensures numerical stability, favoring forget-specific parameters.

3.3 Masking with criticality score

A binary localization mask $m \in \{0,1\}^p$ over all p million trainable parameters is constructed by ranking neurons across all layers, in descending order of criticality score based on our three proposed methods, and greedily selecting neurons until the cumulative parameter count of selected channels meets a budget $\alpha = 30\%$ of total parameters, similarly to [24]. The budget is applied globally, thus the mask is not constrained to act uniformly across layers, so that layers with higher aggregate criticality naturally contribute more selected channels.

3.4 Reinitialization based on criticality score and unlearning

After constructing the mask, we re-initialise each selected weight θ_j using Kaiming normal initialization [13], $2/n_{\text{in}}(j)$, where $n_{\text{in}}(j)$ is the number of input connections to the layer containing θ_j [13]. This matches the initialization used to train the original model and preserves activation variance during the forward pass; unselected weights remain unchanged during re-initialization. The model is then fine-tuned on the retain set D_r for the same number of epochs as in [24]. During fine-tuning, gradients of unselected weights are masked, restricting updates to the re-initialised parameter subset.

4 Experiments

4.1 Dataset

We use the CIFAR-10 dataset [15], which contains 60,000 colour images (32×32) from 10 classes, split into 50,000 training and 10,000 test images. The training set is further divided into 40,000 training and 10,000 validation images, where the validation set is used only for model selection. For the unlearning task, the 40,000 training images are partitioned into a forget set D_f of 5,000 images and a retain set D_r of 35,000 images using a fixed random seed. The forget set comprises 12.5% of the training data and is sampled uniformly at random, preserving the class distribution of the original dataset.

Retain-similar evaluation dataset: A key component of the evaluation strategy is the construction of a *retain-similar set* $D_{r,\text{sim}}$: a subset of the retain set consisting of the retain-set examples most semantically similar to the forget-set examples under the original model's learned representations. Semantic similarity is measured using cosine similarity between 512-dimensional embeddings from the original model f_{θ^o}. Embeddings are computed in a single forward pass for all 5,000 forget-set and 35,000 retain-set samples. For each forget-set sample $x_f \in D_f$, the nearest neighbour is selected from retain-set samples $x_r \in D_r$ of the same class ($y_f = y_r$):

$$x_r^* = \underset{x_r \in \mathcal{D}_r : y_r = y_f}{\arg\max} \text{sim}(x_f, x_r), \tag{10}$$

where $\text{sim}(x_f, x_r)$ is computed by (4). The retain-similar set $D_{r,\text{sim}}$ is formed by collecting the unique nearest neighbours $\{x_r^*\}_{x_r^* \in D_r}$, deduplicated, so that each retain example appears at most once. In practice, the resulting set contains a slightly smaller number of examples than the forget set, since multiple forget-set examples share a nearest neighbour. We used 4,575 samples from $D_{r,\text{sim}}$.

4.2 Model architecture

We use a modified ResNet-18 [14] for all experiments. Because the standard ImageNet configuration aggressively downsamples CIFAR-10 images (32×32), we replace the initial 7×7 stride-2 convolution with a 3×3 stride-1 convolution and remove the first max-pooling layer. We have two models for experiments and for evaluating our proposed method,

- **Original Model** - The original model f_{θ^o} was trained on the full training set using SGD [1] with a learning rate of 0.1, momentum of 0.9, weight decay of 5×10^{-4}, batch size of 128, and 50 epochs. The learning rate followed a cosine annealing schedule [16] with a minimum of 0.001 similar to [24]. The model achieved 89.87% test accuracy. As expected for CIFAR-10, automobile (96.20%) and ship (94.90%) had the highest accuracy, whereas cat (78.30%), dog (84.10%), and bird (85.50%) had the lowest [15].
- **Oracle Model** - The oracle model f_{θ^r} is trained with all the data except the forget set D_f. We use the hyperparameter similar to the original model for training except the number of epochs as 35. The oracle model achieved a test set accuracy of 85.91%, a retain set accuracy of 100%, and a forget set accuracy of 86.32%.

4.3 Evaluation measures

Unlearning quality is evaluated against an oracle model retrained from scratch on D_r, which serves as the gold standard [2]. We report six metrics: test accuracy, retain accuracy, forget accuracy, retain-similar accuracy, membership inference attack (MIA) score [23], and KL divergence [12]. Metrics are reported for the

original model f_{θ^o}, the unlearned model f_{θ^u}, and the oracle model f_{θ^r}. Except for KL divergence, performance is summarized by the delta as follows,

$$\Delta_{\text{metric}} = \text{metric}(f_{\theta^r}) - \text{metric}(f_{\theta^u}), \tag{11}$$

where values closer to zero indicate better agreement with the oracle. Positive values indicate that the unlearned model underperforms the oracle, while negative values indicate over-correction. The retain-similar accuracy delta measures collateral damage to semantically similar retained examples.

Test accuracy measures generalisation to the held-out test set:

$$\Delta_{\text{test}} = \text{acc}_{\text{test}}(f_{\theta^r}) - \text{acc}_{\text{test}}(f_{\theta^u}) \tag{12}$$

A large positive delta indicates collateral damage to the model's general representations.

Retain accuracy measures performance on the full retain set $\mathcal{D}_r$:

$$\Delta_{\text{retain}} = \text{acc}_{\text{retain}}(f_{\theta^r}) - \text{acc}_{\text{retain}}(f_{\theta^u}) \tag{13}$$

A significant positive delta indicates that the unlearning procedure has perturbed parameters important for examples that should be preserved.

Forget accuracy measures whether the model has ceased to correctly classify the forgotten examples:

$$\Delta_{\text{forget}} = \text{acc}_{\text{forget}}(f_{\theta^r}) - \text{acc}_{\text{forget}}(f_{\theta^u}) \tag{14}$$

Following [24], the target is a forget-set accuracy matching the oracle's: too high indicates insufficient forgetting; too low indicates over-forgetting, itself a detectable deviation from the oracle.

Retain-similar accuracy measures performance on $D_{r,\text{sim}}$:

$$\Delta_{\text{rs}} = \text{acc}_{r,\text{sim}}(f_{\theta^r}) - \text{acc}_{r,\text{sim}}(f_{\theta^u}) \tag{15}$$

This is the primary metric for the central hypothesis of our proposal. If retain-aware criticality scoring is effective, Δ_{rs} should be smaller than the corresponding delta produced by the forget-only baseline, indicating that the more conservative channel selection has reduced collateral damage to the retained examples most semantically proximate to the forgotten data.

KL divergence provides a distributional measure of the similarity between the unlearned model and the oracle on the forget set [12, 18]:

$$D_{\text{KL}}(p_{f_{\theta^u}} \parallel p_{f_{\theta^r}}) = \frac{1}{|D_f|} \sum_{x \in D_f} \sum_{y \in Y} p_{f_{\theta^u}}(y \mid x) \log \frac{p_{f_{\theta^u}}(y \mid x)}{p_{f_{\theta^r}}(y \mid x)} \tag{16}$$

where $p_{f_{\theta^u}}(y \mid x)$ and $p_{f_{\theta^r}}(y \mid x)$ are the softmax output distributions of the unlearned and oracle models respectively. A value of zero indicates that the two models are behaviourally indistinguishable on D_f. Unlike accuracy-based metrics, KL divergence captures differences in the full output distribution rather

than just the predicted class, providing sensitivity to subtle residual memorization that accuracy-based metrics may miss [12].

MIA score measures forgetting quality using the confidence-based membership inference attack of [8]. Following their protocol, an RBF-kernel SVC [4, 7] is trained to distinguish retain-set samples (members) from test-set samples (non-members) using the maximum softmax probability as the attack feature. The trained classifier is then applied to the forget set, whose examples are treated as non-members. The MIA score is defined as follows,

$$\mathrm{MIA}_{\mathrm{score}} = \frac{|\{x_f \in D_f : \hat{y}(x_f) = \text{non-member}\}|}{|D_f|}. \tag{17}$$

4.4 Results and Discussion

Table 1 summarizes the delta metrics for the baseline [24] and our proposed method. We subsample data from retain set D_r to compute the criticality score for the retain set.

Method	**Sub**	w	Δ_{test}	Δ_{retain}	Δ_{forget}	Δ_{rs}	Δ_{MIA}	KL
`forget_only(baseline)`	–	–	+2.58	+0.49	+2.70	+0.31	+5.48	0.443
`difference`	**Yes**	–	**+0.64**	+0.11	**+0.56**	+0.04	+4.98	0.395
`difference`	No	–	+1.16	+0.20	+1.24	+0.11	+4.16	**0.380**
`w_diff`	Yes	3	+1.49	+0.45	+1.42	+0.35	+4.30	0.423
`w_diff`	No	3	+1.58	+0.40	+1.96	+0.28	+5.90	0.424
`w_diff`	Yes	5	+2.32	+0.35	+1.96	+0.28	+5.52	0.432
`w_diff`	No	5	+1.67	+0.44	+1.84	+0.33	+6.22	0.429
`w_diff`	Yes	7	+2.42	+0.55	+1.90	+0.33	+5.80	0.413
`w_diff`	No	7	+2.48	+0.48	+2.90	+0.44	+4.58	0.438
`ratio`	Yes	–	−1.16	**+0.01**	−3.86	**+0.02**	−0.34	0.479
`ratio`	No	–	−1.34	**+0.02**	−3.74	**+0.00**	**+0.56**	0.482

Table 1. Unlearning results across baseline and our proposed method: Delta metrics for all eleven experimental conditions. **Sub** denotes subsampled retain set used for criticality computation, Δ_{rs} denotes retain-similar accuracy delta. Positive values indicate the unlearned model performs worse than the oracle, negative values indicate over-correction beyond the oracle, and zero indicates perfect oracle-matching. **Bold letters** indicate the best results.

A random subsample (indicated as column header **Sub**) of 5,000 examples drawn from D_r, while non-subsampled used all 35,000 data in D_r. For our weighted difference method, we choose three values: $w \in \{3, 5, 7\}$, each in both subsampled and non-subsampled variants. A value of $w = 1$ imposes the strictest retain-set protection; large w reduces to the forget-only baseline. The three values tested span a range that is expected to capture the transition between these behaviours: $w = 3$ imposes moderate retain-set protection, $w = 7$ approaches the forget-only regime, and $w = 5$ falls between them.

Discussion of forget-only baseline The `forget_only` baseline has $\Delta_{\text{forget}} = +2.70$, indicating mild over-forgetting relative to the oracle, and $\Delta_{\text{test}} = +2.58$, showing utility loss. Its $\Delta_{\text{rs}} = +0.49$ supports the feature-entanglement hypothesis: forget-only localization selects channels also used by visually similar retain examples, causing collateral damage on $D_{r,\text{sim}}$. The large $\Delta_{\text{MIA}} = +5.48$ and KL divergence of 0.395 further show stronger residual membership signal and poorer oracle alignment than retain-aware methods. Overall, `forget_only` damages retain performance while failing to match the oracle distributionally.

Discussion of similarity aware localized unlearning

Difference method The subsampled `difference` method produces the notable result among our proposed approaches. It achieves a Δ_{forget} of $+0.56$, a Δ_{test} of $+0.64$, a Δ_{retain} $+0.11$, and critically Δ_{rs} of $+0.04$ — the smallest among all conditions that achieves confirms true forgetting. The Δ_{MIA} of $+4.98$ indicating that the unlearned model most closely resembles the oracle from the adversary's perspective. The KL divergence of 0.395 is the second lowest among valid conditions, confirming that the subsampled difference method produces the output distribution on the forget set that is very close to the oracle's — a result that complements and extends the accuracy-based findings.

The non-subsampled difference method remains valid, but its $\Delta_{\text{rs}} = +0.11$ is higher than the subsampled variant. This may reflect the retain set's long-tail structure [9]: full retain aggregation includes atypical examples whose channel usage overlaps with the forget set, shifting the mask away from the forget-similar boundary region. In contrast, subsampling emphasizes denser head-region examples, yielding retain scores better aligned with $D_{r,\text{sim}}$ and therefore lower collateral damage.

Weighted difference method The weighted-difference variants are valid across all values of w, but larger w mainly makes the intervention more aggressive. In the subsampled setting, Δ_{test} increases from $+1.49$ to $+2.42$, while Δ_{MIA} increases from $+4.30$ to $+5.80$, suggesting greater utility loss and residual membership signal as forget-set weighting increases. Subsampling usually improves Δ_{forget}, except at $w = 5$, but its effect on $D_{r,\text{sim}}$ is mixed: non-sampling is better at $w = 3$, while subsampling is better at $w = 5$ and $w = 7$. The KL values remain close overall, indicating similar forget-set distributional alignment. Overall, subsampled $w = 3$ gives the best balance across Δ_{test}, Δ_{forget}, and Δ_{MIA}.

Ratio method The ratio conditions preserve utility but fail to forget. Both variants have strongly negative Δ_{forget} values, -3.86 and -3.74, showing substantial under-forgetting relative to the oracle. Meanwhile, Δ_{test}, Δ_{retain}, and Δ_{rs} remain close to zero or negative, indicating little collateral damage. This is not a favourable trade-off, since the retained utility comes from an insufficiently effective intervention. The likely cause is the ratio criterion $s_j^f/(s_j^r + \epsilon)$, which can over-prioritise channels with small retain scores even when their absolute

forget importance is weak, while downweighting entangled channels that are important to both forget and retain examples. The similar results for subsampled and non-subsampled variants suggest that the failure is intrinsic to the ratio formulation rather than the retain-set size. The elevated KL values further indicate poor oracle alignment on the forget set.

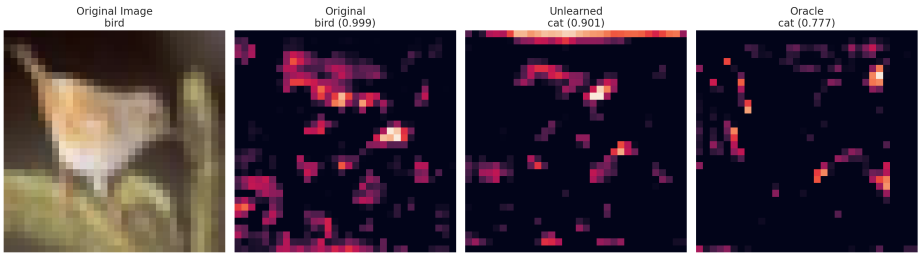


Fig. 2. Forget-set bird. Original (conf. 0.999): correctly classified with sparse scattered activations. Unlearned (conf. 0.901): misclassifies as cat; activation partially smeared, indicating loss of bird-discriminative features. Oracle (conf. 0.777): also misclassifies as cat, confirming the image sits near the bird/cat decision boundary.

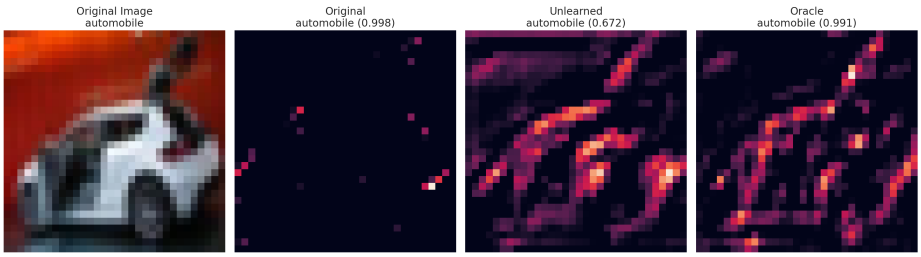


Fig. 3. Test-set automobile. Original (conf. 0.998): near-zero `layer1` activation despite high confidence, indicating classification driven by deeper layers. Unlearned (conf. 0.672): strong structured activation along car edges and body lines. Oracle (conf. 0.991): similarly structured edge-following activation, closely resembling the unlearned model.

Discussion of Δ_{rs} metric Across conditions, Δ_{rs} is generally smaller in magnitude than Δ_{retain}. This can be explained by the construction of $D_{r,\mathrm{sim}}$, which selects retain examples closest to the forget set in embedding space. In our setting, these examples are likely to lie in relatively dense regions with many similar training neighbours, whereas the full retain set also contains rarer, more atypical long-tail examples. Prior work by Feldman et al. [9, 10] shows that memorization is especially important for such rare examples, while high-frequency examples can often be recovered from shared, generalizable features. Consequently, after

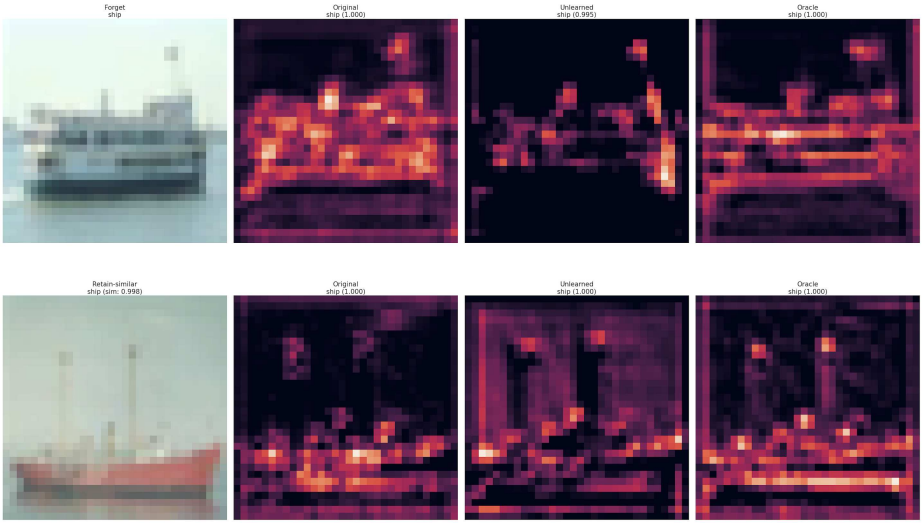


Fig. 4. Forget/retain-similar ship pair (cosine similarity 0.998). Top: unlearning changes the forget-set activation from broad to sparse and localized. Bottom: the retain-similar image remains correctly classified with ship-focused attention, indicating limited collateral damage.

a partial reset and fine-tuning, examples in $D_{r,\mathrm{sim}}$ may be easier to recover than long-tail retain examples, leading to smaller Δ_{rs} than Δ_{retain}. The key point is therefore not that forget-only localization causes large absolute damage to $D_{r,\mathrm{sim}}$, but that it causes more collateral damage than retain-aware methods. The purpose of Δ_{rs} is to measure this excess damage on the retain examples most similar to the forgotten data.

4.5 GradCAM-based Visual analysis

We use Grad-CAM [22] to compare the original, unlearned, and oracle models. We analyze three examples in total: one each from the forget set, test set, and retain-similar set.

Forget-set examples The forget-set bird in Fig. 2 is correctly classified by the original model, which shows broad, scattered activations. Both the unlearned and oracle models predict cat, while the unlearned activation is more diffuse, suggesting disruption of bird-specific features. Its agreement with the oracle indicates oracle-aligned forgetting rather than an unlearning failure.

Test-set examples. The test-set automobile in Fig. 3 is correctly classified by the original model with high confidence (0.998), despite weak Grad-CAM activation at the selected layer. After unlearning, activation becomes more structured along the vehicle's edges and contours, closely resembling the oracle and suggesting recovery of an oracle-like feature representation during retain-set fine-tuning.

Retain-similar examples. Fig. 4 shows a forget-set example and its nearest retain-set neighbour (cosine similarity 0.998). For the forget-set image, the original model exhibits broad activations across the hull and surrounding structure, whereas the unlearned model produces sparser, more localized activations with lower confidence, even than the oracle model. This suggests that unlearning has altered the original representation and reduced reliance on features previously used for classification. For the retain-similar image, the unlearned model still attends to the ship hull and classifies it correctly, indicating that the retained representation is largely preserved. However, its more diffuse activations indicate only partial preservation of the original feature representation.

Computational time Experiments were run on NVIDIA A100 and L40 GPUs, with runtimes reported in MM:SS. Training the original and oracle models took 07:46 and 04:46, respectively. The forget-only baseline required 02:57 in total (00:02 for criticality computation and 02:54 for fine-tuning), while the difference method required 02:59 (00:02 and 02:57), indicating negligible overhead.

5 Conclusion

We introduced a similarity-aware localized unlearning framework that considers parameter importance for both forgotten and retained data. A retain-similar evaluation set was also proposed to measure collateral damage directly. Experiments on CIFAR-10 with ResNet-18 showed that the subsampled difference method achieved the best balance between effective forgetting and retained utility. These results demonstrate that retain-aware localization preserves shared representations more effectively than forget-only localization, with negligible computational overhead. Future work should evaluate the method on larger datasets, architectures, and unlearning settings.

Acknowledgments

M. Kumaran was supported by The University of Manchester and the Manchester Unit of the European Laboratory for Learning and Intelligent Systems (ELLIS). The research of V. Kouni was supported in part by the French National Research Agency under the France 2030 program, reference ANR-23-PEIA-0003. H. Harikumar was supported by the UKRI Turing AI World-Leading Researcher Fellowship (EP/W002973/1), UKRI AI Hub in Generative Models (EP/Y028805/1), European Lighthouse of AI for Sustainability (ELIAS, 10080425), and Laboratory for AI Security Research (LASR). The views expressed in this paper are those of the authors and do not necessarily reflect the position of LASR or His Majesty's Government. The authors would like to acknowledge the assistance by Research IT and the use of the Computational Shared Facility at The University of Manchester.

References

1. Bottou, L.: Stochastic Gradient Descent Tricks. In: Neural networks: tricks of the trade: second edition (2012)
2. Bourtoule, L., Chandrasekaran, V., Choquette-Choo, C.A., Jia, H., Travers, A., Zhang, B., Lie, D., Papernot, N.: Machine Unlearning. In: 2021 IEEE symposium on security and privacy (SP) (2021)
3. Bourtoule, L., Chandrasekaran, V., Choquette-Choo, C.A., Jia, H., Travers, A., Zhang, B., Lie, D., Papernot, N.: Machine unlearning. In: 2021 IEEE symposium on security and privacy (SP). pp. 141–159. IEEE (2021)
4. Broomhead, D.S., Lowe, D.: Radial Basis Functions, Multi-Variable Functional Interpolation and Adaptive Networks. Tech. rep. (1988)
5. Cao, Y., Yang, J.: Towards making systems forget with machine unlearning. In: IEEE Symposium on Security and Privacy (2015)
6. Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U., et al.: Extracting Training Data from Large Language Models. In: 30th USENIX Security Symposium (2021)
7. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning (1995)
8. Fan, C., Liu, J., Zhang, Y., Wong, E., Wei, D., Liu, S.: SalUN: Empowering Machine Unlearning via Gradient-based Weight Saliency in Both Image Classification and Generation. In: International Conference on Learning Representations (2024)
9. Feldman, V.: Does learning require memorization? a short tale about a long tail. In: Proceedings of the 52nd annual ACM SIGACT symposium on theory of computing (2020)
10. Feldman, V., Zhang, C.: What Neural Networks Memorize and Why: Discovering the Long Tail via Influence Estimation. Advances in Neural Information Processing Systems (2020)
11. Fredrikson, M., Jha, S., Ristenpart, T.: Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (2015)
12. Golatkar, A., Achille, A., Soatto, S.: Eternal Sunshine of the Spotless Net: Selective Forgetting in Deep Networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2020)
13. He, K., Zhang, X., Ren, S., Sun, J.: Delving Deep into Rectifiers: Surpassing Human-level Performance on ImageNet Classification. In: Proceedings of the IEEE International Conference on Computer Vision (2015)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016)
15. Krizhevsky, A., Hinton, G., et al.: Learning Multiple Layers of Features from Tiny Images (2009)
16. Loshchilov, I., Hutter, F.: SGDR: Stochastic gradient descent with warm restarts. In: International Conference on Learning Representations (2017)
17. Nguyen, Q.P., Low, B.K.H., Jaillet, P.: Variational Bayesian Unlearning. Advances in Neural Information Processing Systems (2020)
18. Nguyen, T.T., Huynh, T.T., Ren, Z., Nguyen, P.L., Liew, A.W.C., Yin, H., Nguyen, Q.V.H.: A survey of machine unlearning. ACM Transactions on Intelligent Systems and Technology (2025)
19. Pardau, S.L.: The California Consumer Privacy Act: Towards a European-Style Privacy Regime in the United States. Journal of Technology Law & Policy (2018)

20. Raab, C.D.: The Distribution of Privacy Risks: Who Needs Protection? The information society (1998)
21. Schwartz, P.M.: Property, Privacy, and Personal Data. Harvard law review (2004)
22. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization. In: Proceedings of the IEEE International Conference on Computer Vision (2017)
23. Shokri, R., Stronati, M., Song, C., Shmatikov, V.: Membership Inference Attacks Against Machine Learning Models. In: 2017 IEEE symposium on security and privacy (SP) (2017)
24. Torkzadehmahani, R., Nasirigerdeh, R., Kaissis, G., Rueckert, D., Dziugaite, G.K., Triantafillou, E.: Improved Localized Machine Unlearning Through the Lens of Memorization (2024), https://arxiv.org/abs/2412.02432
25. Voigt, P., Von dem Bussche, A.: The EU General Data Protection Regulation (GDPR): A Practical Guide. Springer, Cham (2017)
26. Wei, J., Zhang, Y., Zhang, L.Y., Ding, M., Chen, C., Ong, K.L., Zhang, J., Xiang, Y.: Memorization in Deep Learning: A Survey. ACM Computing Surveys (2025)
27. Xu, J., Wu, Z., Wang, C., Jia, X.: Machine Unlearning: Solutions and Challenges. IEEE Transactions on Emerging Topics in Computational Intelligence (2024)

Appendix



Fig. 5. Training and validation loss (left) and accuracy (right) for the original model $\mathcal{M}(D)$ over 50 epochs. Training loss decreases monotonically to near-zero while validation loss stabilises around 0.33, reflecting the growing train-validation gap characteristic of overfitting to training examples including $\mathcal{D}_f$. Validation accuracy plateaus around 90% from epoch 35 onward with minor fluctuations. The dashed line marks the best checkpoint (model) at epoch 46.

Fig. 5 shows the training and validation curves for the original model. The original model $\mathcal{M}(D)$ was trained for 50 epochs in the entire 40 , 000 sample training subset and achieved a test set accuracy of 89.87%, consistent with expected performance for a well-tuned ResNet-18 in CIFAR-10 [14]. The training trajectory followed the expected pattern: rapid initial learning in the first ten epochs, a slower refinement phase as the cosine learning rate schedule decayed, and training accuracy saturating near 100% by epoch 39 while validation accuracy continued to improve gradually. The best model was saved at epoch 46. The original model achieves 100% accuracy on both the retain and forget sets, as expected for a fully converged model evaluated on its own training data. Table 2 shows the per-class test accuracy for the original model, oracle model, and our unlearned model (based on subsampled `difference`). The largest drops are in cat (9.4 pp) and bird (7.7 pp) — the two most visually similar and hardest CIFAR-10 classes [15] — suggesting the unlearning intervention disproportionately disrupts discriminative features near class boundaries. Visually distinctive classes such as automobile (2.1 pp) and ship (1.5 pp) suffer the smallest drops, consistent with their features being more separable and therefore less affected by the channel reset.

Class	Original (%)	Unlearned (%)	Oracle (%)	$\Delta_{\text{org,unlearn}}$ (pp)	$\Delta_{\text{oracle,unlearn}}$ (pp)
Airplane	90.90	87.00	88.50	+3.90	+1.50
Automobile	96.20	94.10	93.60	+2.10	−0.50
Bird	85.50	77.80	79.60	+7.70	+1.80
Cat	78.30	68.90	67.40	+9.40	−1.50
Deer	89.30	84.10	87.10	+5.20	+3.00
Dog	84.10	80.10	77.90	+4.00	−2.20
Frog	93.10	89.10	90.90	+4.00	+1.80
Horse	93.20	89.00	89.30	+4.20	+0.30
Ship	94.90	93.40	93.00	+1.50	−0.40
Truck	93.20	89.20	91.80	+4.00	+2.60
Overall	89.87	85.27	85.91	4.60	0.64

Table 2. Per-class test set accuracy for the original model, the subsampled `difference` unlearned model, and the oracle model. $\Delta_{\text{org,unlearn}}$ (pp) denotes the difference in percentage points between the original and unlearned model. $\Delta_{\text{oracle,unlearn}}$ (pp) denotes the difference in percentage points between the oracle and unlearned model. The difference is largest for cat and bird between the unlearned model and original model; the unlearned model closely tracks the oracle, suggesting effective unlearning.

Method	**Sub**	**w**	**Crit time**	**Finetune time**	**Total time**
Original model	–	–	–	–	07:46
Oracle model	–	–	–	–	04:46
Baseline (del)	–	–	00:02	02:54	02:57
`Ours (difference)`	Yes	–	00:02	02:57	02:59
`Ours (difference)`	No	–	00:06	02:57	03:03
`Ours (weighted)`	Yes	3	00:02	02:54	02:56
`Ours (weighted)`	No	3	00:06	02:55	03:01
`Ours (weighted)`	Yes	5	00:02	02:53	02:55
`Ours (weighted)`	No	5	00:06	02:55	03:01
`Ours (weighted)`	Yes	7	00:02	02:57	02:59
`Ours (weighted)`	No	7	00:06	02:58	03:04
`Ours (ratio)`	Yes	–	00:02	02:58	03:00
`Ours (ratio)`	No	–	00:05	02:56	03:02

Table 3. Wall-clock time (MM:SS, minutes and seconds, rounded to the nearest second) taken per pipeline stage for all eleven experimental conditions. Sub denotes whether a subsampled retain set was used for criticality computation. Original is the from-scratch training of the base model; Oracle is the retain-only retraining used as the gold-standard reference; Crit is the criticality-scoring step alone; and Finetune is the subsequent finetuning stage. Original and Oracle have no unlearning pipeline, so only total training time is shown.

Table 3 reports wall-clock times for all experimental conditions. The criticality-scoring step is negligible in all cases, taking at most six seconds even without subsampling, meaning the total pipeline cost is dominated almost entirely by

finetuning. All unlearning methods complete in under three minutes and five seconds, compared to seven minutes and forty-six seconds for original training and four minutes and forty-six seconds for oracle retraining. Subsampling reduces criticality computation time from roughly five–six seconds to two seconds across all variants, with no meaningful effect on finetuning time, making it a cost-free efficiency gain.

Method	**Sub**	**Finetuning Size**	Δ_{test}	Δ_{retain}	Δ_{forget}	Δ_{rs}	Δ_{MIA}	KL
`forget_only`	–	500	+65.11	+78.76	+64.80	+77.79	+11.12	1.812
`forget_only`	–	1000	+48.75	+62.45	+49.60	+61.09	+10.98	1.484
`forget_only`	–	2000	+35.25	+47.04	+34.16	+43.98	−12.74	1.089
`forget_only`	–	5000	+22.79	+33.73	+21.60	+30.67	−30.18	0.802
`forget_only`	–	10000	+13.96	+19.94	+13.24	+16.90	−11.06	0.625
`forget_only`	–	20000	+5.91	+8.48	+6.72	+7.32	+0.88	0.479
`forget_only`	–	35000	+2.58	+0.49	+2.70	+0.31	+5.48	0.443
`difference`	Yes	500	+54.48	+68.19	+55.54	+65.84	−38.50	1.692
`difference`	Yes	1000	+48.52	+61.40	+48.82	+59.02	−60.10	1.572
`difference`	Yes	2000	+37.06	+49.12	+37.28	+45.68	−47.32	1.316
`difference`	Yes	5000	+26.62	+34.83	+26.32	+31.06	−32.72	0.995
`difference`	Yes	10000	+13.49	+19.54	+14.22	+15.91	−14.64	0.643
`difference`	Yes	20000	+4.68	+7.78	+5.28	+5.99	−0.62	0.441
`difference`	Yes	35000	+0.64	+0.11	+0.56	+0.04	+4.98	0.395

Table 4. Effect of finetuning retain set size on unlearning quality. Sub denotes subsampled retain set used for criticality computation, Finetuning size denotes the size of the retain set used for finetuning

Table 4 examines how the size of the retain set used during finetuning affects unlearning quality, comparing the `forget_only` baseline against the subsampled `difference` method. Both methods exhibit a consistent trend: smaller finetuning sets yield larger deviations from the oracle across all metrics, with deltas decreasing monotonically as retain set size grows toward 35,000. At 35,000 samples, the `difference` method achieves the smallest overall deltas, with $\Delta_{\text{test}} = 0.64$, $\Delta_{\text{forget}} = 0.56$, and $\Delta_{\text{rs}} = 0.04$, closely tracking the oracle on all retain-side metrics. These results indicate that finetuning set size is a key determinant of unlearning quality, and that the full retain set of 35,000 samples is necessary to achieve oracle-level performance. Table 5 shows the channel selection produced by our subsampled `difference` method, which is used as the representative condition. The selection is markedly non-uniform across layers: layer3 has the majority of its channels selected (up to 229/256 in `layer3.0.downsample.0`), reflecting the criticality scoring concentrating forget-set importance in the deeper, more class-specific residual stages. By contrast, `layer4.1.conv2` contributes zero selected channels despite its depth — a consequence of the differential criterion: its parameters are sufficiently important to the retain set that the difference score is zeroed out entirely. Earlier layers show moderate, more evenly

Layer	Total channels	Selected	Fraction (%)
`conv1`	64	31	48.4
`layer1.0.conv1`	64	40	62.5
`layer1.0.conv2`	64	46	71.9
`layer1.1.conv1`	64	27	42.2
`layer1.1.conv2`	64	49	76.6
`layer2.0.conv1`	128	94	73.4
`layer2.0.conv2`	128	101	78.9
`layer2.0.downsample.0`	128	106	82.8
`layer2.1.conv1`	128	71	55.5
`layer2.1.conv2`	128	78	60.9
`layer3.0.conv1`	256	228	89.1
`layer3.0.conv2`	256	226	88.3
`layer3.0.downsample.0`	256	229	89.5
`layer3.1.conv1`	256	115	44.9
`layer3.1.conv2`	256	97	37.9
`layer4.0.conv1`	512	305	59.6
`layer4.0.conv2`	512	116	22.7
`layer4.0.downsample.0`	512	371	72.5
`layer4.1.conv1`	512	58	11.3
`layer4.1.conv2`	512	0	0.0
`fc`	10	10	100.0
Total parameters	3,352,873 / 11,173,962 (30.01%)		

Table 5. Channels selected per layer under the subsampled `difference` method with $\alpha = 30\%$ parameter budget. Selection is highest in layer3 (up to 89.5%) where forget-set criticality is concentrated, and unevenly distributed in layer4: `layer4.0.downsample.0` has 72.5% of channels selected while `layer4.1.conv2` has none, reflecting the differential criterion zeroing out channels whose retain-set importance meets or exceeds their forget-set importance. The fully connected layer is selected in its entirety, indicating that all output logit weights are implicated in the forget-set predictions.

distributed selection, consistent with their role in encoding lower-level features shared across both forget and retain examples. Fig. 6 visualises how each metric varies with w across both subsampled and non-subsampled conditions, treating the unweighted `difference` method as $w = 1$. The figure reveals several consistent patterns. On the retain-similar metric, the subsampled `difference` method at $w = 1$ achieves the lowest delta of all conditions, and performance degrades as w increases toward the baseline, confirming that stronger forget-set weighting erodes retain-set protection. The non-subsampled variant consistently underperforms its subsampled counterpart on Δ_{rs}. The test accuracy panel shows a similar ordering at $w = 1$, with the subsampled `difference` method achieving the best scores before degrading as w increases.

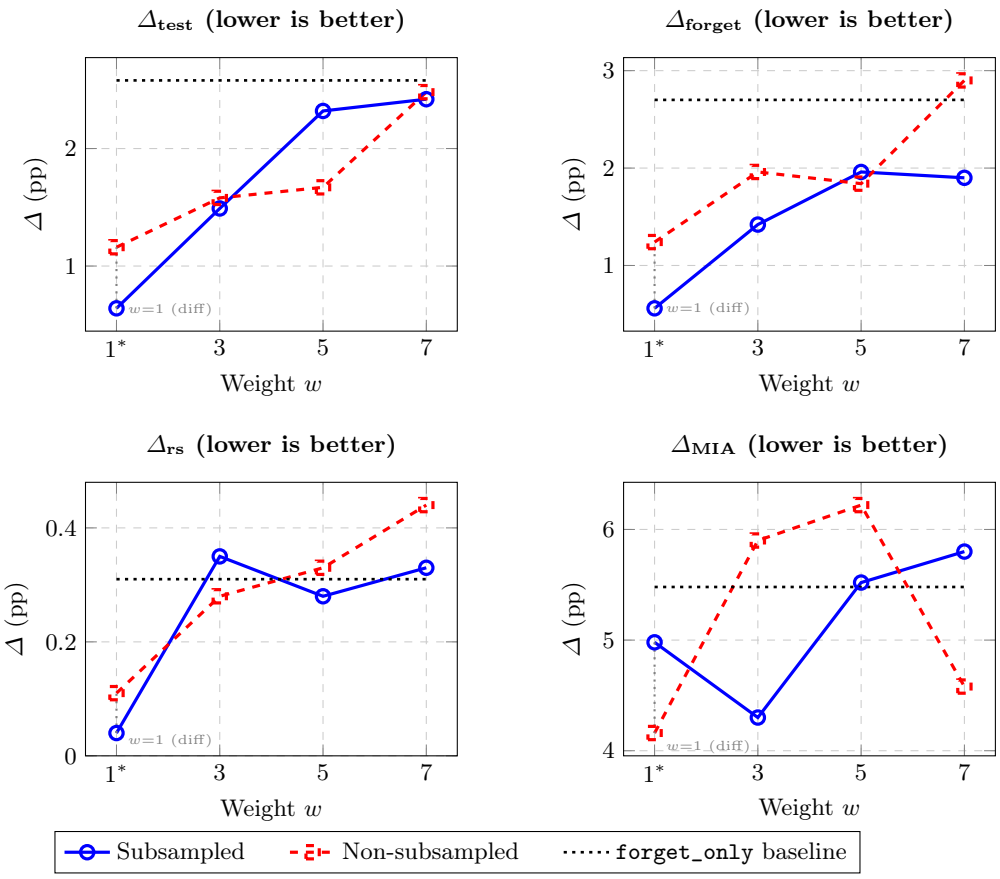


Fig. 6. Metric deltas as a function of weight w for subsampled and non-subsampled conditions. The `difference` method is treated as $w = 1$ (marked 1^* on the x-axis). The black dotted horizontal line marks the `forget_only` baseline score on each metric as a reference. Each metric is reported as Δ = oracle−unlearned; lower values indicate closer oracle-matching. The dotted vertical segment at $w = 1$ highlights the gap between the two subsampling conditions at the unweighted case.