Pull-to-Outlier & Contrastive Objective-level (POCO) Unlearning: A Framework for Sample and Objective Forgetting

Anonymous authors
Paper under double-blind review

Abstract

Current Machine Unlearning (MU) methods require full retraining or extensive fine-tuning, lack formal removal criteria, and focus only on sample-level forgetting, limiting their practicality. We address these gaps with two lightweight, projection-only techniques operating above frozen feature extractors. Pull-to-Outlier Unlearning (POU) offers a transparent, unsupervised geometric removal method by displacing embeddings of unwanted samples or entire classes into synthetic outlier regions, while preserving downstream performance and distilling knowledge of the remaining data. To the best of our knowledge, Contrastive Objective-level Unlearning (COU) is the first method to remove learned objectives. It perturbs projection weights to eliminate a target objective's influence. Then it realigns the original data manifold after objective perturbation. We validate POU on CIFAR10, CIFAR100, and Caltech-256 with ResNet-based backbones, showing efficient instance and class forgetting with minimal impact on retained accuracy. COU is tested on DINO and CLIP feature representations, providing evidence for objective-level removal in the studied settings while preserving non-target structure.

1 Introduction

Machine unlearning (MU) refers to a family of techniques that enable the selective removal of specific training data or learned behaviors from a trained machine learning model, without the need to retrain the model from scratch. The motivation for MU arises from the growing legal, ethical, and operational requirements in real-world deployments. For instance, the European Union's General Data Protection Regulation (GDPR), particularly Article 17—the "right to erasure"—mandates that individuals must be able to request the removal of their personal data from any system that has used it (European Parliament and Council, 2016). Similarly, the California Consumer Privacy Act (CCPA) enforces users' rights to request deletion of their information, creating a pressing need for compliant and practical unlearning mechanisms in AI systems (California State Legislature, 2018).

Beyond legal mandates, MU is increasingly essential in sensitive deployment contexts. In cybersecurity, models must revoke the influence of adversarially poisoned or malicious data that could compromise the safety of predictions. However, studies reveal that traditional unlearning techniques often fail to fully mitigate such attacks, underscoring the need for more resilient strategies (Pawelczyk et al., 2024). In healthcare analytics, institutions must ensure that withdrawn patient records no longer influence clinical predictions, especially when consent is revoked after the data have already been used (Sakib & Xie, 2024). Federated learning scenarios present additional challenges: individual clients may request removal of their contributions after global aggregation, necessitating efficient and non-disruptive unlearning methods (Wu et al., 2023). These examples collectively highlight that MU is not only a theoretical safeguard but a practical necessity for deploying AI in dynamic, privacy-sensitive environments.

Despite this progress, existing MU methods share four key limitations:

1. They primarily focus on *sample-level* or *class-level* removal, offering no systematic mechanism to forget an entire learned objective or capability embedded within a model's representation.

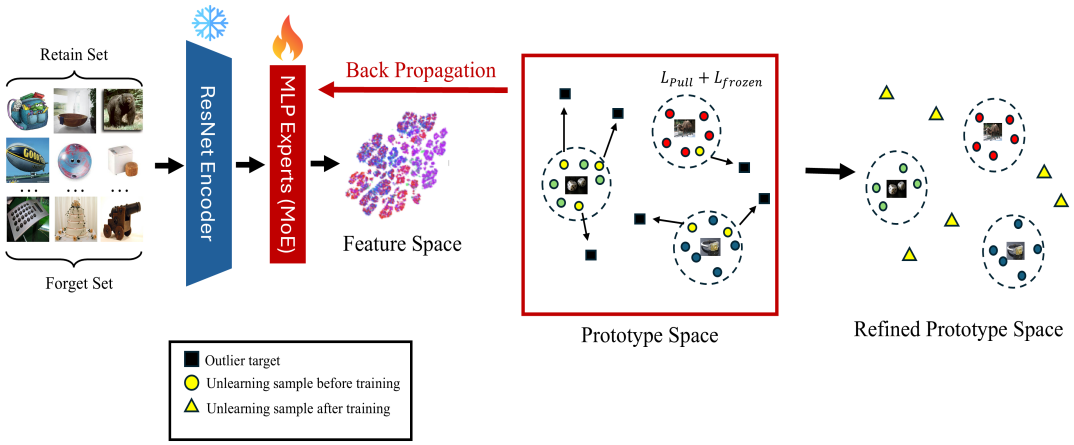


Figure 1: Pipeline of the proposed Pull-to-Outlier Unlearning (POU) method. The process begins with feature extraction using a frozen encoder, followed by prototype selection via clustering. Unlearned samples are then mapped to synthetic outlier targets using expert-specific projection heads, while a frozen loss preserves the structure of retained samples. Final outputs are aggregated for evaluation.

2. They do not provide a transparent, per-example verification mechanism to determine whether unlearning has been successfully achieved. As a result, many methods rely on statistical heuristics or indirect metrics to assess success.
3. Existing methods typically require full or partial fine-tuning of the backbone network, whereas our projection-only approach avoids backbone updates entirely, offering significant computational advantages.
4. Only a limited number of approaches, such as Label-Agnostic Forgetting (LAF), support fully *unsupervised* forgetting, where the method can operate without class labels or supervision during the unlearning process (Shen et al., 2024).

To address all four of these challenges, we propose two lightweight unlearning methods that operate entirely above frozen feature extractors and, therefore, avoid any modification to the pretrained backbone:

- **Pull-to-Outlier Unlearning (POU)** is designed for scalable, unsupervised forgetting of specific samples or entire classes. It works by pushing the embeddings of targeted samples outside the known data manifold, making them geometrically incompatible with downstream use. POU leverages a mixture-of-experts (MoE) framework: retained embeddings are first clustered, and each cluster is anchored by a prototype. For every sample to be unlearned, a small residual MLP "expert" is assigned and trained to map it to a synthetic outlier vector located beyond the global min–max bounds. To preserve the rest of the representation space, a frozen-loss term keeps all other embeddings near their original positions. This modular design enables scalable deletion across large sets without updating the backbone, while also offering a transparent test, based on geometric displacement, to verify that the unlearning has succeeded.
- **Contrastive Objective-level Unlearning (COU)** addresses a higher-level goal: the removal of an entire learned capability, such as a class or behavior, rather than individual samples. It begins with a projection head trained using supervised contrastive learning to encode task-specific semantics. COU then selectively perturbs this projection to collapse the representation of the target objective, effectively erasing its contribution. A lightweight fine-tuning step on retained data restores class separability for all remaining tasks. Unlike prior instance-based or class-based methods, COU performs this forgetting entirely through projection weights, enabling objective-level unlearning in the studied settings without modifying the feature extractor. This makes it a lightweight approach for selective projection-space capability removal.

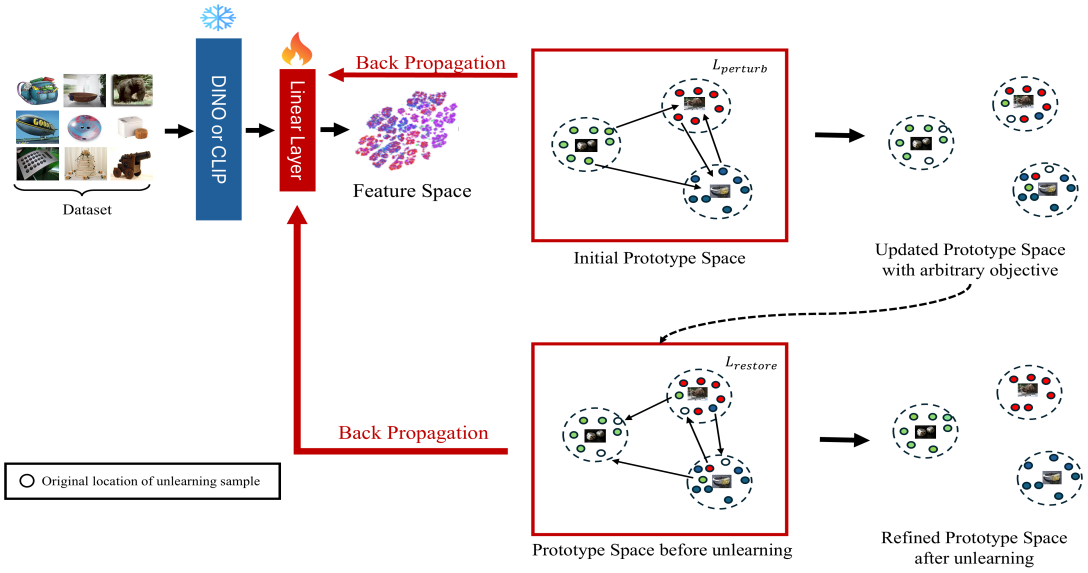


Figure 2: Pipeline of the proposed Contrastive Objective-level Unlearning (COU) method. A projection head is first trained on top of frozen backbone features using supervised contrastive loss. After a new objective is externally introduced by modifying class associations, COU unlearns this objective by pulling affected embeddings back to their original positions, restoring the original structure of the representation space.

We use *task* for a dataset-defined prediction problem (e.g., CIFAR10 classification), and *objective* (or *capability*) for a learned projection-space behavior that we aim to erase.

2 Related Work

2.1 Machine Unlearning

Early efforts in machine unlearning focused on exact removal for simple models. One such foundational approach proposed an algorithm for summation-based learners that could analytically revoke the influence of specific training points, but this technique did not extend to modern deep architectures (Cao & Yang, 2015). While full retraining remains the most accurate method to ensure data removal, it is computationally impractical for large-scale models and datasets. To mitigate this, partition-based approaches like SISA (Sharded, Isolated, Sliced Aggregation) training were introduced, which divide the training data into multiple disjoint subsets. When a removal request is issued, only a subset of these partitions require retraining, significantly reducing computational burden while maintaining fidelity (Bourtoule et al., 2019). Another line of work approximates the unlearning process by reversing stochastic gradient descent updates. UnrollSGD simulates unlearning by backtracking the most recent optimization steps, enabling partial reversibility of training with bounded approximation error (Thudi et al., 2021).

In contrast to these strategies, some methods attempt to remove information by adjusting model gradients directly. One such method, commonly referred to as NegGrad, applies updates in the negative gradient direction of the samples to forget. While effective in some cases, this approach can cause undesirable shifts in the representation of non-target samples, especially when fine-tuning deep backbones. More recently, unsupervised strategies have emerged. Label-Agnostic Forgetting (LAF) addresses the removal of entire classes without relying on class labels. It uses a variational distribution matching mechanism to suppress the influence of target distributions, enabling scalable and label-free forgetting in deep neural networks (Shen et al., 2024).

Recent works have explored contrastive learning objectives to improve machine unlearning. For instance, one recent work proposes pushing embeddings of the data to be forgotten away from their original classes and toward alternative representations, modifying InfoNCE-style objectives (Lee et al., 2024). Another work

introduces gradient-based constraints to diminish the influence of selected data points in contrastive and supervised learning settings, requiring only a few fine-tuning steps (Wang & Chen, 2024). A further study focuses on the auditing aspect by calibrating alignment metrics within contrastive models to verify successful removal without full retraining (Wang et al., 2024). While these methods offer improvements at the sample level, they still require backbone updates and do not support objective-level unlearning.

Summary of trade-offs. Prior MU approaches occupy different points in the trade-off between fidelity, cost, and auditability. **Partition-based methods** (e.g., SISA) reduce retraining cost by localizing updates, at the expense of storing multiple shard models and retraining affected partitions on each request. **Gradient-based approximations** (e.g., NegGrad) are lightweight to implement but often destabilize non-target representations, especially when updates propagate through the backbone. **Trajectory reversal** (e.g., UnrollSGD) can approximate removal with bounded error assumptions but requires storing optimization trajectories and incurs additional second-order computation. **Unsupervised class removal** methods (e.g., LAF) reduce reliance on labels but introduce extra auxiliary models (e.g., generative components) and additional training phases. In contrast, **POCO** aims to (i) avoid backbone updates by operating projection-only above frozen features, (ii) provide a simple, example-level geometric verification signal for POU (outlier displacement under calibrated thresholds), and (iii) extend unlearning beyond samples by enabling projection-level objective removal with COU; these design choices trade a small head-level overhead for reduced end-to-end training and simpler deployment when deletion requests are frequent.

2.2 Prototypical Learning

Prototype-based methods represent each class with a centroid in the embedding space, offering interpretability (Angelov & Soares, 2019; Angelov et al., 2025; Chen et al., 2018; Rymarczyk et al., 2021) and robustness, especially in few-shot learning (Snell et al., 2017). Prototypical Networks classify by measuring distances to these centroids. Clustering-based approaches like DeepCluster alternate between k-means (Lloyd, 1982) and classification via pseudo-labels (Caron et al., 2018), while SwAV uses online cluster assignments to enable large-scale self-supervised learning without labels (Caron et al., 2020).

These prototype-based strategies also support open-set recognition and anomaly detection by modeling in-distribution regions. Our POU method leverages this by (i) using fixed prototypes as anchors for generating synthetic outliers, and (ii) preserving retained samples via a frozen-loss objective—thus maintaining the data manifold while selectively removing targets.

2.3 Outlier Detection

Outlier detection identifies data points that deviate from the underlying distribution. Classical methods include z-scores, Tukey's fences, LOF for density comparison (Breunig et al., 2000), and DBSCAN for detecting sparse regions (Ester et al., 1996). With deep learning (Pang et al., 2020b), approaches, such as one-class SVMs (Hearst et al., 1998; Schölkopf et al., 1999), Autoencoders (Rumelhart et al., 1986), and GANs (Goodfellow et al., 2014) have been widely adopted for high-dimensional settings. Surveys by Chandola et al. and Pang et al. highlight this shift toward learned representations (Chandola et al., 2009; Pang et al., 2020a).

POU builds on these ideas by generating synthetic outlier targets beyond global feature bounds and using σ-band checks around cluster prototypes to transparently verify that unlearned embeddings lie outside valid class regions.

3 Methodology

3.1 Pretraining and Feature Extraction

For the POU, we begin by pretraining a ResNet-18 backbone (He et al., 2015) followed by a three-layer residual MLP classifier using standard cross-entropy loss on the full training set. The residual MLP acts as a lightweight head and forms the projection module for POU. During unlearning, we continue updating this

same MLP head while keeping the ResNet backbone frozen. This setup ensures that the learned embedding space remains stable, and the unlearning process is confined to the projection layer alone. Since POU is designed to operate in a label-free manner, this pretraining step serves to construct a semantically meaningful feature space that remains decoupled from the downstream forgetting mechanism.

For COU, we use pretrained feature extractors from DINOv2 (Caron et al., 2021; Oquab et al., 2023) and CLIP (Radford et al., 2021) ViT-L/14 to obtain high-level representations of input images. On top of these frozen vision transformer backbones, we train a linear projection head using supervised contrastive loss. This loss encourages intra-class compactness and inter-class separation in the projected embedding space, establishing the structure needed for selective objective-level removal. The linear head is later updated to remove the influence of a specific objective, while the backbone remains unchanged. Throughout the rest of the paper, we denote the frozen encoder output for input x_i by h_i and the corresponding projected embedding after the task-specific head by z_i; clustering, prototype selection, and unlearning operate on z_i.

3.2 Initial Prototype Space

Given frozen encoder outputs $\{h_i\}_{i=1}^N$ and their corresponding projected embeddings $\{z_i\}_{i=1}^N \subset \mathbb{R}^d$ after the task-specific head, we apply K-means clustering to the projected embeddings to partition them into C clusters. Let $\{\mu_c\}_{c=1}^C$ denote the resulting cluster centers, and let $c(i) \in \{1, \ldots, C\}$ be the cluster assignment for embedding z_i.

To obtain representative samples, we define the prototype p_c for cluster c as the sample closest to its center:

$$p_c = \arg \min_{i \,:\, c(i)=c} \left\| z_i - \mu_c \right\|_2.$$

These prototypes are retained and fixed throughout the unlearning phase to provide semantic anchors in the embedding space.

For POU, prototype selection is performed in the embedding space of the pretrained ResNet-18 followed by a three-layer residual MLP. These prototypes are used both to define synthetic outlier targets (based on global min-max bounds) and to verify forgetting via σ-band outlier detection. In COU, the prototypes are selected from embeddings obtained using a linear projection head trained on frozen DINOv2 or CLIP ViT-L/14 features with supervised contrastive loss. Here, prototypes serve to define task-specific semantics and guide perturbation during the objective removal phase. This shared prototype space provides a consistent geometric reference for both sample-level and objective-level forgetting.

3.3 Pull-to-Outlier Unlearning (POU)

Pull-to-Outlier Unlearning (POU) removes the influence of specific samples by relocating their embeddings beyond the semantic boundary of the retained dataset. The method updates only the projection head, using a combination of geometric target generation and two complementary loss terms. Below, we describe each stage in detail.

3.3.1 Outlier Target Generation

Let $\{z_i\}_{i=1}^N \subset \mathbb{R}^D$ be the projected embeddings of all training samples, and let $\mathcal{U}$ denote the set of indices to be unlearned. We first compute per-dimension bounds over the retained set $\mathcal{N} = \{1, \ldots, N\} \setminus \mathcal{U}$:

$$g_{\min,d} = \min_{j \in \mathcal{N}} z_{j,d}, \quad g_{\max,d} = \max_{j \in \mathcal{N}} z_{j,d}, \quad \text{for } d = 1, \ldots, D.$$

Then, for each $i \in \mathcal{U}$, we define a synthetic outlier target $\tau_i \in \mathbb{R}^D$:

$$\tau_{i,d} = \begin{cases} g_{\min,d} - \delta, & \text{with probability } 0.5, \\ g_{\max,d} + \delta, & \text{otherwise,} \end{cases}$$

where $\delta > 0$ is a fixed margin. This ensures that τ_i lies outside the retained embedding manifold in all dimensions.

3.3.2 Pull Loss

We use a pull loss to push each unlearning sample z_i toward its corresponding outlier target τ_i:

$$L_{\text{pull}} = \frac{1}{|\mathcal{U}|} \sum_{i \in \mathcal{U}} \|z_i - \tau_i\|_2^2.$$

This displaces the embeddings of forgotten samples away from high-density regions of the training distribution.

3.3.3 Frozen Loss

To maintain stability of the non-target data, we penalize movement of retained embeddings relative to their original positions $z_j^{(0)}$:

$$L_{\text{frozen}} = \frac{1}{|\mathcal{N}|} \sum_{j \in \mathcal{N}} \|z_j - z_j^{(0)}\|_2^2.$$

This regularization preserves the semantic structure of the remaining dataset.

3.3.4 Combined Objective

The total loss is a weighted combination:

$$L_{\text{POU}} = \lambda_{\text{pull}} \cdot L_{\text{pull}} + \lambda_{\text{frozen}} \cdot L_{\text{frozen}}.$$

Only the projection head is optimized.

3.3.5 Outlier Detection Criterion

To provide a transparent *geometric* check that unlearned embeddings have exited the retained clusters, we check whether the final embedding of each unlearned sample lies outside all prototype regions. For each prototype p_c, we compute per-dimension standard deviation over samples X_c assigned to that cluster:

$$\sigma_{c,d} = \sqrt{\frac{1}{|X_c|} \sum_{x_i \in X_c} (z_{i,d} - \mu_{c,d})^2}.$$

We then assess whether z_i deviates from μ_c in a sufficient number of dimensions:

$$|\{d : |z_{i,d} - \mu_{c,d}| > 3 \cdot \sigma_{c,d}\}| > \tau \cdot D.$$

This 3-sigma criterion serves as a statistical boundary to detect whether a sample has exited the semantic region defined by any cluster. If the condition holds for all c, the sample is flagged as successfully forgotten. This provides an interpretable, prototype-aware certificate of *geometric displacement*. In Appendix Table 7, we empirically compare this 3σ-based criterion against standard alternatives (nearest-center distance and Mahalanobis distance) under matched false-positive rates on retained data.

All retained and test samples are classified using a nearest-prototype classifier, computed by assigning each embedding to the closest prototype with Euclidean distance.

3.3.6 Expert Aggregation Strategy

To scale unlearning to large deletion sets while preserving frozen backbone features, POU adopts a Mixture-of-Experts (MoE) design (Jacobs et al., 1991). The forget set is split into disjoint subsets, each handled by a separate residual MLP expert initialized from the pretrained projection head and trained independently using pull and frozen losses.

For retained samples, we apply soft aggregation by averaging outputs from all experts, promoting stability. For unlearned samples, we adopt hard expert routing (Fedus et al., 2021), selecting the expert whose output best matches the sample's outlier target. This design enables scalable, modular unlearning without modifying the backbone.

3.4 Contrastive Objective-level Unlearning (COU)

Contrastive Objective-level Unlearning (COU) enables the removal of entire learned objectives or capabilities through projection-only updates. Unlike conventional sample-wise unlearning approaches, COU suppresses a complete representational function by first perturbing the projection space and then restoring it without modifying the backbone network. This process unfolds in two distinct phases.

3.4.1 Learning with Arbitrary Objective

We begin by selecting a subset of training samples $\{x_i\}_{i \in \mathcal{U}}$ and assigning each to a pseudo-target class, chosen as the second-nearest prototype p_c such that $c \neq y_i$. This defines an arbitrary, synthetic objective in the latent space that deviates from the model's original class structure. To impose this new objective, we optimize a perturbation loss that maximizes alignment between the sample and its pseudo-target:

$$L_{\text{perturb}} = \frac{1}{|\mathcal{U}|} \sum_{i \in \mathcal{U}} \left[1 - \max_{c \neq y_i} \cos(z_i, p_c) \right],$$

where z_i denotes the projected embedding obtained from the frozen encoder output h_i of x_i. This updates the projection head to encode the injected objective, leaving all other parts of the model untouched.

3.4.2 Unlearning

To remove the influence of the modified objective and revert to the original semantic geometry, we apply a contrastive pull-back mechanism. For each perturbed sample, we restore its embedding to its original location $z_i^{(0)}$ recorded prior to the perturbation phase. The unlearning loss is defined as:

$$L_{\text{restore}} = \frac{1}{|\mathcal{U}|} \sum_{i \in \mathcal{U}} \left[1 - \cos(z_i, z_i^{(0)}) \right].$$

Importantly, no additional preservation terms are used for non-target samples, making the method simple, scalable, and focused entirely on forgetting the injected objective.

COU may offer a valuable mechanism for self-improving or agent-driven learning systems, where dynamic objectives are proposed, explored, and revised over time. In such settings, COU may be useful for retracting undesired or unstable objectives introduced by agents, enabling selective unlearning on top of foundational models through lightweight, head-only updates.

4 Experiments

4.1 Datasets

We evaluate our proposed unlearning methods on three standard image classification benchmarks. For **CIFAR10** (Krizhevsky, 2009), POU is applied to the full dataset of 60,000 images across 10 classes, while COU uses a subsample of 10,000 images (1,000 per class) for fine-grained control over learned objectives. For **CIFAR100** (Krizhevsky, 2009), we use the complete set of 60,000 images across 100 fine-grained categories to test POU under greater class granularity and clustering density. **Caltech-256** (Griffin et al., 2007) includes 30,607 images from 256 diverse categories and is used exclusively for POU to assess unlearning in high-variability domains with complex prototype and semantic structures. For all datasets, we split 90% of the data for training and 10% for testing.

4.2 Evaluation Metrics

To evaluate both the effectiveness of unlearning and the preservation of retained knowledge, we adopt distinct metrics for instance-level unlearning (POU) and objective-level unlearning (COU). For POU, we report: (1) **Remaining Set Accuracy**, the classification accuracy on retained samples, where higher scores indicate preserved knowledge; (2) **Unlearning Set Accuracy**, the accuracy on deleted samples, where values near

0% reflect effective forgetting; (3) **Test Set Accuracy**, measuring performance on the original test set after unlearning; (4) **Trustworthiness** (Venna & Kaski, 2001), which evaluates local neighborhood consistency between the original and final embeddings of retained data. Concretely, let $X = \{x_i\}_{i=1}^n$ denote the original retained embeddings and $Y = \{y_i\}_{i=1}^n$ the corresponding embeddings after unlearning. Let $\mathcal{N}_k^Y(i)$ be the set of k nearest neighbors of y_i in Y, and let $r_X(i,j)$ be the rank of x_j among all points in X when sorting by increasing distance to x_i. Define the "intrusion" set

$$\mathcal{U}_k(i) \;=\; \mathcal{N}_k^Y(i) \setminus \mathcal{N}_k^X(i).$$

The trustworthiness at neighborhood size k is

$$\mathrm{TW}(k) \;=\; 1 - \frac{2}{nk(2n-3k-1)} \sum_{i=1}^{n} \sum_{j \in \mathcal{U}_k(i)} \big(r_X(i,j) - k\big),$$

where higher values indicate better local-structure preservation. We report TW(5) in all experiments; and (5) L_2 **Drift**, the average Euclidean distance between the original and updated retained embeddings, quantifying collateral change.

For COU, we compute: (1) **Trustworthiness (Post-Perturbation & Post-Unlearning)**, calculated over all samples to assess the projection space's structural changes and recovery; and (2) L_2 **Drift (Post-Perturbation & Post-Unlearning)**, representing embedding displacement induced by perturbation and reduced through unlearning.

4.3 Baselines

Retrain: The most intuitive unlearning baseline involves full model retraining on the retained dataset after the removal of target samples. While it provides an exact removal guarantee, it is computationally infeasible in practice and scales poorly to frequent deletions.

NegGrad: This method approximates forgetting by fine-tuning the model with reversed gradients on the forget set. For COU, we report results using NegGrad as the sole baseline, although it was not originally designed for objective-level unlearning and lacks explicit mechanisms for forgetting latent objectives or projection-specific behavior.

Unrolling SGD (Thudi et al., 2021): UnrollSGD leverages optimization trajectory inversion by tracking and reversing gradient steps associated with the target samples. It achieves approximate unlearning with bounded error but requires storing past gradients and optimizer states, making it memory-intensive.

SISA (Bourtoule et al., 2019): Sharded, Isolated, Sliced, and Aggregated (SISA) training splits the dataset into multiple shards and slices to localize retraining when unlearning is required.

Label-Agnostic Forgetting (LAF) (Shen et al., 2024): A recent unsupervised method that removes entire classes via variational distribution matching without relying on explicit labels. LAF demonstrates that principled removal is possible even without supervision.

4.4 Implementation Details

All experiments were conducted on a machine equipped with an AMD Ryzen 9 5900HX CPU, 32 GB RAM, and an NVIDIA RTX 3080 Laptop GPU with 16 GB memory.

We use a batch size of 64 and a learning rate of 1e−4 across all experiments. All baseline methods are trained for 100 epochs, while the ResNet-18 + MLP backbone used in POU and Retrain is pretrained for 120 epochs. For **POU**, we use the Adam optimizer and set $\lambda_{\text{pull}} = 1.0$, $\lambda_{\text{frozen}} = 50.0$, and margin $\delta = 20.0$. We apply early stopping when both the pull loss value drops below 10 and the frozen loss falls below 0.02, indicating sufficient displacement of unlearned samples and stability of retained embeddings. For **COU**, we also use Adam and train the linear projection head for 5 epochs using a supervised contrastive loss with temperature 0.07, and stop unlearning once the mean L_2 drift of pulled-back samples falls below 2.

Table 1: Instance-level unlearning results on CIFAR10, CIFAR100, and Caltech-256. We randomly subsample 1% of the training set as the unlearning set and use the same set across all baselines. Metrics include: **Rem** (Remaining Set Accuracy ↑), **Unl** (Unlearning Set Accuracy ↓), **Test** (Test Set Accuracy ↑), **TW** (Trustworthiness ↑), and $\boldsymbol{L_2}$ **Drift** (Average $\boldsymbol{L_2}$ Embedding Drift ↓). Arrows indicate whether higher (↑) or lower (↓) values are better.

Method	**CIFAR10**					**CIFAR100**					**Caltech-256**				
	Rem ↑	Unl ↓	Test ↑	TW ↑	L_2 ↓	Rem ↑	Unl ↓	Test ↑	TW ↑	L_2 ↓	Rem ↑	Unl ↓	Test ↑	TW ↑	L_2 ↓
Retrain	99.62%	86.80%	88.19%	-	-	99.24%	59.00%	61.98%	-	-	99.30%	47.01%	46.24%	-	-
NegGrad	10.02%	7.60%	10.00%	0.5335	$> 2 \times 10^8$	1.00%	0.80%	1.00%	0.5130	$> 7.9 \times 10^8$	0.65%	0.00%	0.57%	0.5123	$> 5.5 \times 10^8$
UnrollSGD	99.44%	93.08%	87.03%	0.9900	1.3633	98.78%	99.80%	60.91%	0.9931	1.3466	99.42%	100.00%	43.45%	0.9884	6.7948
SISA	99.52%	100.00%	88.01%	0.9941	7.2353	99.30%	96.90%	61.70%	0.9992	6.0514	86.86%	80.59%	45.28%	0.9816	7.6901
LAF	86.43%	83.20%	74.62%	0.4995	3.5835	57.31%	54.40%	34.34%	0.5008	6.1496	60.73%	36.07%	46.40%	0.4989	3.6107
POU (Ours)	99.07%	0.00%	86.89%	1.0000	0.9451	98.58%	0.00%	61.41%	1.0000	1.3388	98.25%	0.00%	39.02%	1.0000	0.8585

For the baselines: **SISA** uses 5 shards, 2 slices per shard, and 20 epochs per slice. **LAF** is trained with 20 VAE epochs and 5 unlearning epochs. **NegGrad** uses $\lambda_{\text{neg}} = 1.0$ for gradient ascent on the forget set. **UnrollSGD** performs 5 epochs of SGD trajectory recording and applies influence reversal over 50 unroll steps.

4.5 Computational and Memory Cost

POU performs unlearning by updating only the projection head on top of a frozen feature extractor. In our implementation, the unlearning stage operates on cached 512-d features and optimizes a 3-layer residual MLP; each expert head contains $4 \cdot (512^2 + 512) = 1{,}050{,}624$ parameters (about 4 MB in FP32). In all main experiments we use K=5 experts; sensitivity to K (via chunk size) and other hyperparameters is reported in Appendix Table 8. With the Mixture-of-Experts design, storing K=5 heads adds a small footprint (about 20 MB in FP32) relative to a full backbone. At inference, retained samples are evaluated by averaging K expert outputs, and forgotten samples are routed to the closest expert by matching their outlier target; this adds K lightweight head forward passes but does not require backbone recomputation.

In contrast, retraining-based and gradient-based baselines incur end-to-end compute through the full image model. **Retrain** and **NegGrad** backpropagate through the backbone for many epochs/steps on 224×224 images, making their cost dominated by repeated full-network training. **UnrollSGD** additionally records optimization trajectories and uses second-order computations (e.g., Hessian–vector products) during reversal, increasing both compute and auxiliary state. **SISA** trains and stores multiple shard models and retrains affected partitions on deletion requests, multiplying storage and training effort by the number of shards/slices. **LAF** trains additional generative models (e.g., VAEs) and performs further optimization to suppress the forget distribution, adding extra networks and epochs beyond standard fine-tuning. Overall, POU trades a small, explicit MoE head overhead for avoiding repeated backbone updates and large auxiliary training state.

4.6 Instance and Class-level Unlearning Results

We evaluate instance-level and class-level forgetting using POU across CIFAR10, CIFAR100, and Caltech-256. As shown in Table 1 and Table 2, POU consistently achieves perfect forgetting on the unlearned subset (0% accuracy), while preserving accuracy on the remaining set and test set. In contrast, prior baselines struggle to eliminate the target information without causing significant collateral damage to the rest of the model. Notably, the slight accuracy drop of POU on the remaining and test sets compared to Retrain may stem from the difference in classification strategy, as POU uses a nearest-prototype classifier instead of a standard softmax classification head; however, its perfect trustworthiness score indicates that the underlying representation structure is fully preserved.

Trustworthiness and L_2 drift metrics further reveal that POU maintains the integrity of the retained representation space. These claims are visually supported in Figure 4, where POU preserves compact cluster structure while cleanly displacing unlearned samples to synthetic outlier regions. Similarly, Figure 3 confirms

Table 2: Class-level unlearning results on CIFAR100 and Caltech-256 using POU and baseline methods. We consistently select the 5th class for unlearning across all experiments.

Method	**CIFAR100**					**Caltech-256**				
	Rem ↑	Unl ↓	Test ↑	TW ↑	L_2 ↓	Rem ↑	Unl ↓	Test ↑	TW ↑	L_2 ↓
Retrain	99.03%	0.00%	61.39%	-	-	98.43%	0.00%	44.42%	-	-
NegGrad	1.00%	0.00%	1.00%	0.5349	$> 8.2 \times 10^4$	0.57%	0.00%	0.65%	0.5117	$> 5.2 \times 10^8$
UnrollSGD	99.24%	97.26%	50.10%	0.9887	6.7679	98.52%	96.33%	44.83%	0.9884	6.7584
LAF	80.45%	100.00%	46.35%	0.5022	2.1888	98.19%	98.89%	45.03%	0.4976	4.3002
POU (Ours)	98.58%	0.00%	60.99%	1.0000	0.9662	98.27%	0.00%	39.02%	1.000	0.9609

that even when entire classes are forgotten, the semantic organization of remaining data remains intact. We further probe (i) augmentation-closure for instance unlearning, (ii) collateral impact on semantically related classes under class unlearning, and (iii) adversarial recoverability; results are summarized in Appendix Section A.2.

We also evaluated robustness using a Membership Inference Attack (MIA) (Table 3). For Attack A (remain vs. test), methods like UnrollSGD reduce leakage closer to chance, while POU maintains stable behavior for retained data, ensuring representation preservation. For Attack B (forget vs. test), POU is clearly the best, driving leakage down to near-chance (2.89%) and thus providing the strongest guarantee of effective forgetting. This highlights that POU achieves the best balance between utility and privacy.

Table 3: Threshold-based Membership Inference Attack (MIA) AUC (%): Attack A = remain vs. test; Attack B = forget vs. test. Lower is better.

Method	**AUC-A Before**	**AUC-A After**	**AUC-B Before**	**AUC-B After**
POU	59.34%	59.75%	60.06%	**2.89%**
NegGrad	61.51%	50.26%	61.69%	49.05%
unroll_sgd	63.01%	49.68%	63.46%	49.96%
LAF	61.79%	58.82%	63.95%	62.72%
SISA	62.94%	62.91%	61.98%	61.84%

4.7 Objective-level Unlearning Results

In objective-level unlearning, the goal is to remove a latent capability encoded by the projection head without modifying the frozen backbone. Table 4 reports results for the synthetic objective injection setting. In this setting, we first train a 1024→512 linear projection head on frozen DINOv2 or CLIP features using supervised contrastive loss, then randomly select 100 training samples and induce an artificial objective by optimizing the head so that these samples are pulled toward their second-nearest K-means prototype in cosine similarity rather than their original class-aligned region. We additionally evaluate COU on an organically learned *spurious supervision objective*, induced by continuing supervised-contrastive training under partially corrupted supervision. Specifically, on frozen DINOv2 features we select 500 samples, relabel them with balanced incorrect targets, continue supervised-contrastive training on this corrupted supervision signal, and then apply COU by restoring embeddings toward the clean reference projection. As shown in Table 5, COU restores the original neighborhood structure (trustworthiness returns to 1.0) and substantially reduces embedding drift. A qualitative t-SNE visualization is provided in Appendix Figure 6.

NegGrad, by contrast, shows an inability to reverse the impact of the learned objective, resulting in high drift and a loss of semantic separability. Figure 5 visualizes this difference: COU successfully restores original class-wise clusters after unlearning, while NegGrad collapses them into entangled or flattened manifolds. These results indicate that COU supports capability-level forgetting via projection-only tuning in the synthetic and spurious-supervision settings studied here.

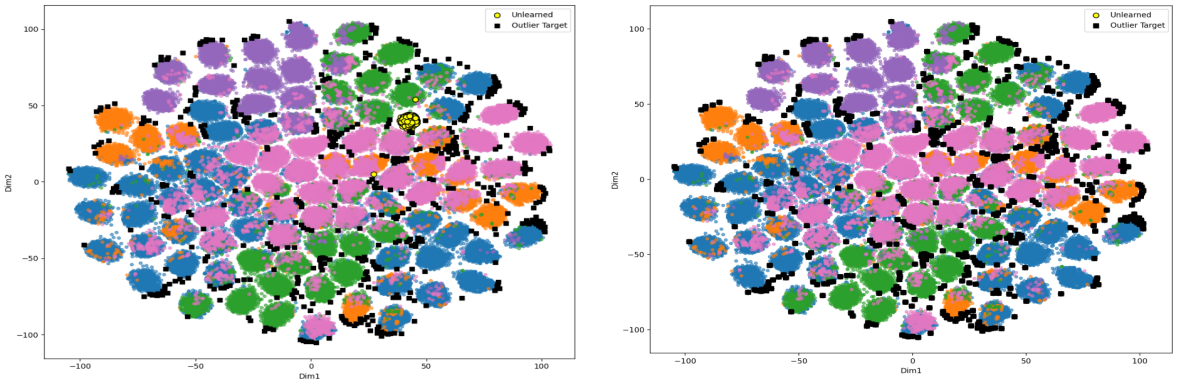


Figure 3: t-SNE visualization of class-level unlearning on CIFAR100 using POU. **Left:** The original projection space before unlearning shows well-separated latent subclusters for each class. **Right:** After unlearning, the deleted class samples (yellow) are successfully displaced to synthetic outlier regions (black squares), while the rest of the latent structure remains stable. This confirms that POU achieves class forgetting without disturbing non-target data.

Table 4: Objective-level unlearning results on a subset of CIFAR10 using COU and NegGrad, evaluated on frozen DINOv2 and CLIP ViT-L/14 features. Metrics: Trustworthiness (TW) after perturbation and unlearning phases, and L_2 Drift after unlearning.

Method	**DINOv2 ViT-L/14**			**CLIP ViT-L/14**		
	TW Pert	TW Unlearn ↑	L_2 ↓	TW Pert	TW Unlearn ↑	L_2 ↓
NegGrad	0.9747	0.9699	201.3660	0.9845	0.9607	93.3761
COU (Ours)	0.9747	1.0000	1.6496	0.9845	0.9999	0.8286

4.8 Discussion

4.8.1 Ablations

To understand the role of each component in POU, we conduct an ablation study focused on the frozen loss, as reported in Table 6. When the frozen loss is removed, forgetting becomes incomplete and unlearned samples remain entangled with retained data. Although performance on the retained set may appear high, the underlying structure is destabilized, leading to high drift and reduced trustworthiness.

In contrast, incorporating the frozen loss consistently maintains the geometric alignment of retained embeddings while allowing targeted forgetting of unwanted samples. This term proves essential for balancing deletion and preservation objectives within the same embedding space.

4.8.2 Visual Analysis of Structural Preservation

Figures 3, 5, and 4 provide qualitative evidence supporting the numerical results. In instance-level unlearning, POU visibly displaces unlearned samples far from their original clusters without disrupting the surrounding distribution. This demonstrates that forgetting is both complete and localized.

For objective-level unlearning, COU recovers the original projection layout after removing the arbitrary objective, confirming that projection-space perturbations are reversible under our design. In contrast, NegGrad leads to significant semantic collapse and deformation. These visualizations underscore the importance of geometric regularity and prototype alignment in designing effective unlearning mechanisms.

Table 5: COU on an organically learned spurious supervision objective (CIFAR10 subset with frozen DINOv2 ViT-L/14 features). We report trustworthiness (TW) and mean L_2 drift between the clean SupCon embeddings and the embeddings after spurious supervision training / after COU unlearning. The "affected subset" denotes samples trained under spurious supervision.

Backbone	$\mathrm{TW}_{\mathrm{spur}} \uparrow$	$\mathrm{TW}_{\mathrm{COU}} \uparrow$	L_2 (all) $\downarrow$	L_2 (affected subset) $\downarrow$
DINOv2 ViT-L/14	0.9753	1.0000	17.8204 $\rightarrow$ 2.4276	22.2129 $\rightarrow$ 2.4437

Table 6: Ablation study of the POU method. We compare the full method with the variant that removes the L_{frozen} term across CIFAR10, CIFAR100, and Caltech-256. Metrics: Rem ($\uparrow$), Unl ($\downarrow$), Test ($\uparrow$), TW ($\uparrow$), and L_2 Drift ($\downarrow$).

(a) CIFAR10

Setting	Rem $\uparrow$	Unl $\downarrow$	Test $\uparrow$	TW $\uparrow$	L_2 $\downarrow$
POU without L_{frozen}	99.23%	22.45%	86.31%	0.9971	168.78
POU with L_{frozen}	99.07%	0.00%	86.89%	1.0000	0.95

(b) CIFAR100

Setting	Rem $\uparrow$	Unl $\downarrow$	Test $\uparrow$	TW $\uparrow$	L_2 $\downarrow$
POU without L_{frozen}	93.95%	16.34%	58.22%	0.9972	197.18
POU with L_{frozen}	98.58%	0.00%	61.41%	1.0000	1.34

(c) Caltech-256

Setting	Rem $\uparrow$	Unl $\downarrow$	Test $\uparrow$	TW $\uparrow$	L_2 $\downarrow$
POU without L_{frozen}	75.47%	7.82%	26.23%	0.9820	329.42
POU with L_{frozen}	98.25%	0.00%	39.02%	1.0000	0.86

5 Broader Impact

POCO targets practical machine unlearning under deployment constraints (e.g., privacy-driven deletion requests) by enabling projection-only updates that can reduce repeated full retraining. As a dual-use capability, efficient unlearning can also be misused to conceal undesirable training influence, so deployments should pair it with access controls, logging, and independent auditing (e.g., membership/privacy tests) and clearly state the scope/limitations of any verification signal.

6 Conclusion

We introduced POCO, a unified framework for sample-level and objective-level machine unlearning through projection-only techniques. Our POU method enables efficient instance and class forgetting by geometrically displacing unwanted embeddings, while COU removes entire objectives by perturbing the projection space and then recovering its original structure. Both methods operate above frozen feature extractors, offering scalable and interpretable forgetting without retraining. Extensive evaluations on CIFAR10, CIFAR100, and Caltech-256 confirm that POCO achieves high retention, minimal drift, and reliable forgetting. These results demonstrate the potential of our approach for enabling efficient, interpretable unlearning in privacy-preserving and adaptable machine learning systems.

References

Plamen Angelov, Dmitry Kangin, and Ziyang Zhang. Ideal: Interpretable-by-design algorithms for learning from foundation feature spaces. *Neurocomputing*, 626:129464, 2025. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2025.129464.

Plamen P. Angelov and Eduardo Almeida Soares. Towards explainable deep neural networks (xdnn). *Neural networks : the official journal of the International Neural Network Society*, 130:185–194, 2019.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. *2021 IEEE Symposium on Security and Privacy (SP)*, pp. 141–159, 2019.

Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. Lof: identifying density-based local outliers. pp. 93–104. Association for Computing Machinery, 2000. ISBN 1581132174. doi: 10.1145/342009.335388.

California State Legislature. California Consumer Privacy Act of 2018. California Legislative Information, 2018.

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.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *European Conference on Computer Vision*, 2018.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *ArXiv*, abs/2006.09882, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Herv'e J'egou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9630–9640, 2021.

Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3), 2009. doi: 10.1145/1541880.1541882.

Chaofan Chen, Oscar Li, Alina Jade Barnett, Jonathan Su, and Cynthia Rudin. This looks like that: deep learning for interpretable image recognition. In *Neural Information Processing Systems*, 2018.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. AAAI Press, 1996.

European Parliament and Council. Regulation (EU) 2016/679 (General Data Protection Regulation). Official Journal of the European Union, 2016.

William Fedus, Barret Zoph, and Noam M. Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *ArXiv*, abs/2101.03961, 2021.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *Neural Information Processing Systems*, 2014.

Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.

Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2015.

M.A. Hearst, S.T. Dumais, E. Osuna, J. Platt, and B. Scholkopf. Support vector machines. *IEEE Intelligent Systems and their Applications*, 13(4):18–28, 1998. doi: 10.1109/5254.708428.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991. doi: 10.1162/neco.1991.3.1.79.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Hong Kyu Lee, Qiuchen Zhang, Carl Yang, Jian Lou, and Li Xiong. Contrastive unlearning: A contrastive approach to machine unlearning. *ArXiv*, abs/2401.10458, 2024.

S. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982. doi: 10.1109/TIT.1982.1056489.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Q. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russ Howes, Po-Yao (Bernie) Huang, Shang-Wen Li, Ishan Misra, Michael G. Rabbat, Vasu Sharma, Gabriel Synnaeve, Huijiao Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision. *ArXiv*, abs/2304.07193, 2023.

Guansong Pang, Chunhua Shen, Longbing Cao, and Anton van den Hengel. Deep learning for anomaly detection. *ACM Computing Surveys (CSUR)*, 54:1 – 38, 2020a.

Guansong Pang, Chunhua Shen, Longbing Cao, and Anton van den Hengel. Deep learning for anomaly detection. *ACM Computing Surveys (CSUR)*, 54:1 – 38, 2020b. URL `https://api.semanticscholar.org/CorpusID:220363945`.

Martin Pawelczyk, Jimmy Z. Di, Yiwei Lu, Gautam Kamath, Ayush Sekhari, and Seth Neel. Machine unlearning fails to remove data poisoning attacks. *ArXiv*, abs/2406.17216, 2024.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, October 1986. ISSN 1476-4687. doi: 10.1038/323533a0.

Dawid Rymarczyk, Lukasz Struski, Michal G'orszczak, Koryna Lewandowska, Jacek Tabor, and Bartosz Zieli'nski. Interpretable image classification with differentiable prototypes assignment. In *European Conference on Computer Vision*, 2021.

Shahnewaz Karim Sakib and Mengjun Xie. Machine unlearning in digital healthcare: Addressing technical and ethical challenges. *Proceedings of the AAAI Symposium Series*, 4(1):319–322, November 2024. ISSN 2994-4317. doi: 10.1609/aaaiss.v4i1.31809.

Bernhard Schölkopf, Robert C Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. In S. Solla, T. Leen, and K. Müller (eds.), *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999.

Shaofei Shen, Chenhao Zhang, Yawen Zhao, Alina Bialkowski, Weitong Tony Chen, and Miao Xu. Label-agnostic forgetting: A supervision-free unlearning in deep models. In *The Twelfth International Conference on Learning Representations*, 2024.

Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. In *Neural Information Processing Systems*, 2017.

Anvith Thudi, Gabriel Deza, Varun Chandrasekaran, and Nicolas Papernot. Unrolling sgd: Understanding factors influencing machine unlearning. *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*, pp. 303–319, 2021.

Jarkko Venna and Samuel Kaski. Neighborhood preservation in nonlinear projection methods: An experimental study. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN 2001)*, pp. 485–491. Springer, 2001.

Yihan Wang, Yiwei Lu, Guojun Zhang, Franziska Boenisch, Adam Dziedzic, Yaoliang Yu, and Xiao-Shan Gao. Alignment calibration: Machine unlearning for contrastive learning under auditing. *ArXiv*, abs/2406.03603, 2024.

Zixin Wang and Kongyang Chen. Machine unlearning in contrastive learning. *ArXiv*, abs/2405.07317, 2024.

Chen Wu, SENCUN ZHU, and Prasenjit Mitra. Unlearning backdoor attacks in federated learning. In *ICLR 2023 Workshop on Backdoor Attacks and Defenses in Machine Learning*, 2023.

A Appendix

A.1 Outlier Criterion Comparison for POU Verification

We compare the proposed 3σ-based displacement check to two standard alternatives: (i) a quantile-based threshold on the nearest-cluster-center L_2 distance, and (ii) a minimum Mahalanobis distance to clusters using shrinkage covariance (Ledoit–Wolf). For a fair comparison, each detector threshold is calibrated on retained embeddings to match a fixed false-positive rate (FPR = 1%), and we report the resulting true-positive rate (TPR) on unlearned samples in Table 7.

A.2 Robustness Probes: Augmentation Consistency, Related Classes, and Adversarial Recoverability

We report three robustness probes for POU: (i) augmentation consistency for instance unlearning, (ii) the impact on semantically related classes under class unlearning, and (iii) adversarial recoverability in image space. Outlier thresholds are calibrated to 1% FPR on retained training data.

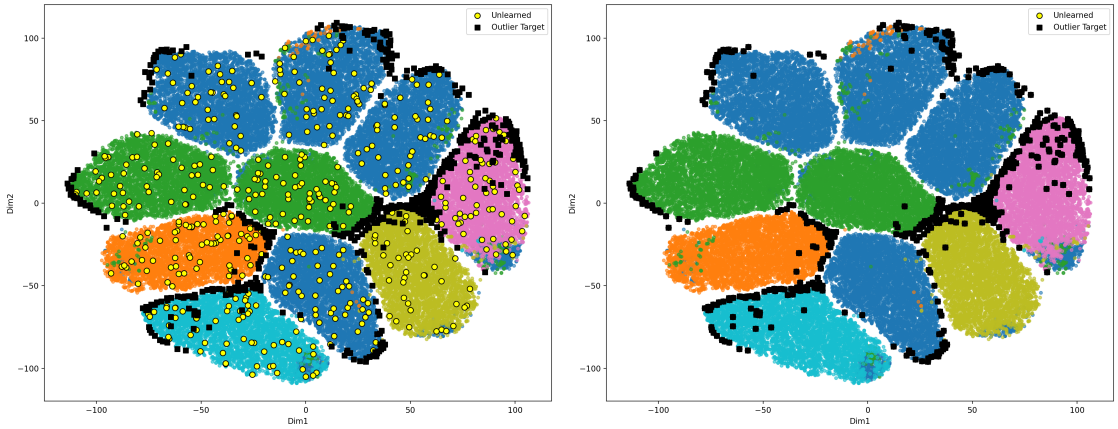
Augmentation consistency (instance unlearning). On CIFAR-10 instance unlearning, we generate 10 random augmentations (crop/rotation/flip/color jitter) for each forgotten training image (500×10 probes) and evaluate whether the augmented variants are also rejected as outliers. After unlearning, the outlier true-positive rate (TPR) on augmented probes is 33.56% using the 3σ-fraction detector and 6.88% using nearest-center L_2. These results indicate that forgetting transfers only partially to augmented views, with the 3σ-based detector capturing this effect more reliably than nearest-center distance.

Semantically related classes (class unlearning). On CIFAR-100 class unlearning ("big cats" example), we unlearn a target class (lion) and evaluate a set of related classes (tiger/leopard). After unlearning, forgotten-class test samples are flagged as outliers with TPR 69% (3σ) and 66% (L_2). Related classes show a 4% outlier rate under both criteria and maintain 64.50% test accuracy. This suggests that POU removes the target class while largely preserving semantically neighboring classes, indicating limited collateral damage.

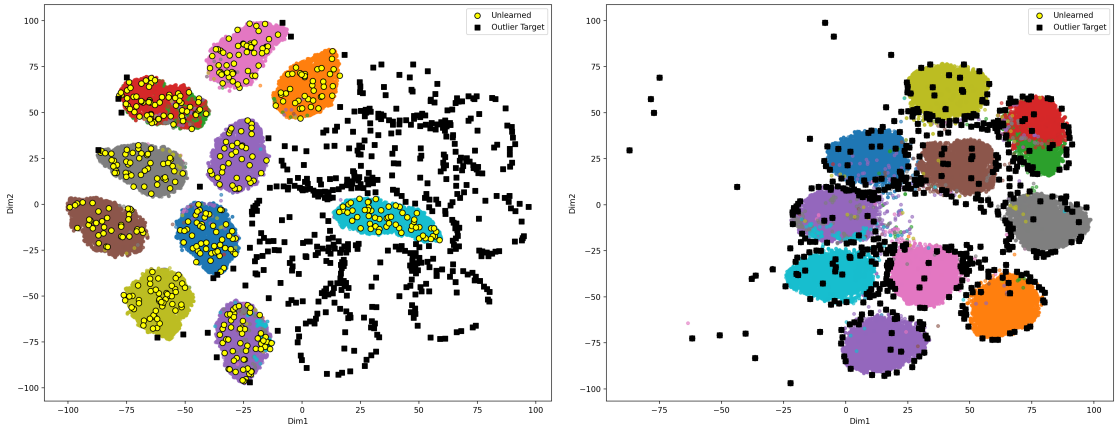
Adversarial recoverability (image-space PGD). On CIFAR-10 instance unlearning, we apply a Projected Gradient Descent (PGD) L_∞ attack to 50 forgotten training images (20 steps; $\epsilon \in \{2, 4, 8\}/255$), optimizing to reduce the outlier score and increase true-class confidence. The rate of re-entering the in-distribution region (outlier score $\leq$ threshold) is 44%, 60%, and 34% at $\epsilon = \{2, 4, 8\}/255$, respectively, while true-label recovery remains low at 2%, 0%, and 0%. These results suggest that although adversarial perturbations can sometimes move forgotten samples back toward the retained region geometrically, they rarely recover the original semantic identity, indicating that the forgetting effect remains difficult to reverse.

A.3 Hyperparameter Ablations for POU

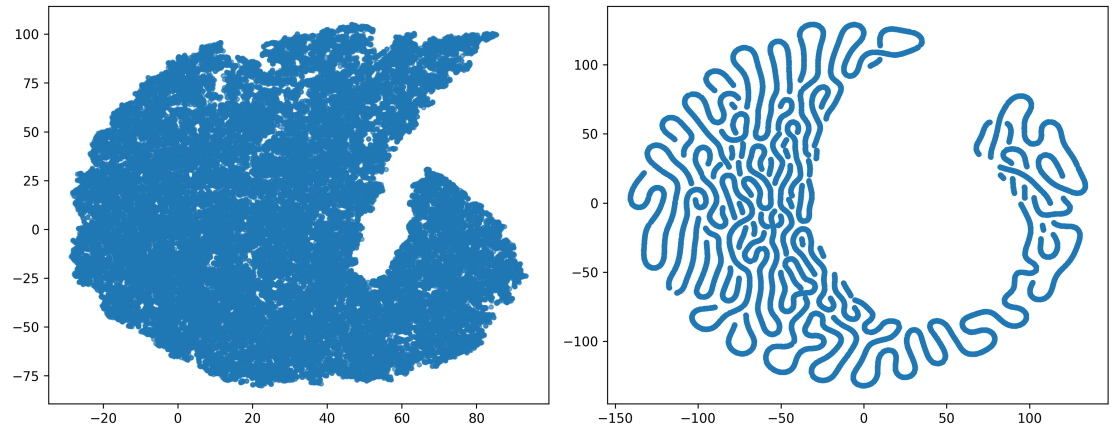
We ablate key POU hyperparameters on CIFAR-10 instance unlearning (1% forget set, 500 samples) while keeping the backbone frozen. All settings achieve perfect forgetting (0% accuracy on the unlearned set); the main effect is on representation preservation, reflected by trustworthiness and retained-set L_2 drift. Results are provided in Table 8.



(a) POU with frozen loss

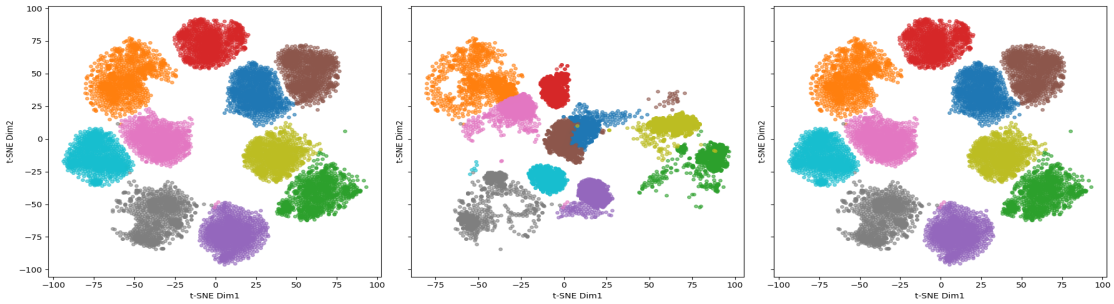


(b) POU without frozen loss

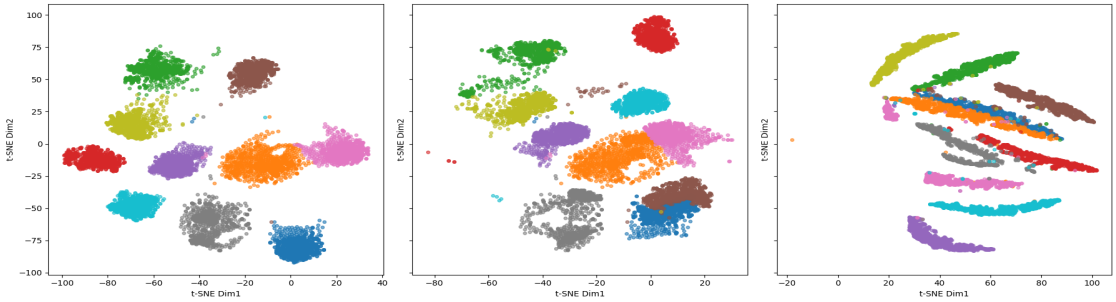


(c) NegGrad

Figure 4: t-SNE visualizations of instance-level unlearning on CIFAR10. The comparison illustrates the importance of the frozen loss in POU. With frozen loss (top), cluster structures are preserved and unlearned samples are cleanly displaced to outlier regions. Without frozen loss (middle), forgetting is incomplete and retained clusters are distorted. NegGrad (bottom) shows severe representation collapse and poor separation.



(a) COU: t-SNE visualizations before and after objective-level unlearning on DINOv2 features. Left: original embedding space. Middle: after injecting arbitrary objective. Right: after COU unlearning. COU restores clean cluster structure.



(b) NegGrad: t-SNE visualizations before and after objective-level unlearning on DINOv2 features. Left: original embedding space. Middle: after injecting arbitrary objective. Right: after NegGrad-based unlearning. Class structure is not recovered.

Figure 5: Qualitative comparison of COU and NegGrad for objective-level unlearning on DINOv2 embeddings. Only COU restores the original semantic structure of the projection space after removing injected objectives.

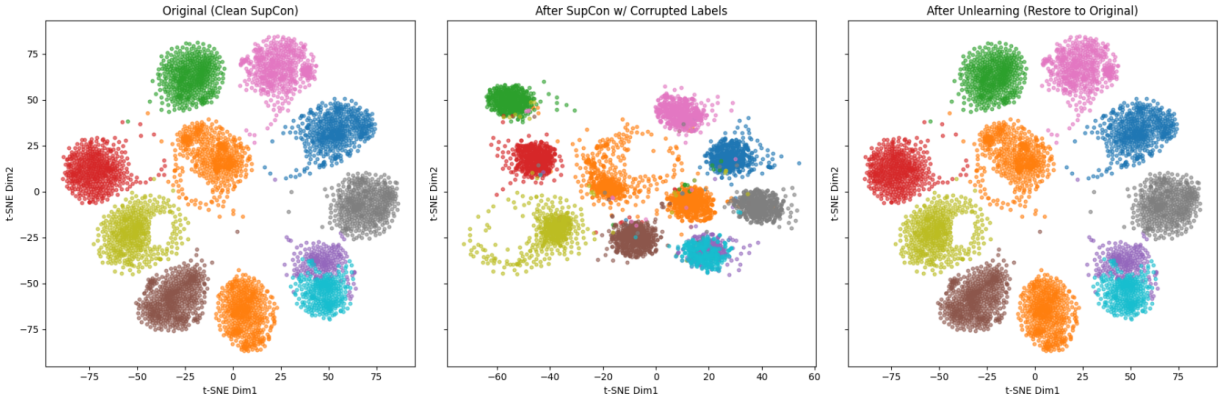


Figure 6: t-SNE visualization for COU under an organically learned spurious supervision objective (CIFAR10 subset with frozen DINOv2 ViT-L/14 features). **Left:** clean SupCon projection space. **Middle:** after continued contrastive training under spurious supervision, which distorts the projection geometry. **Right:** after COU unlearning, which restores the original structure.

Table 7: Comparison of outlier detection criteria used as a geometric displacement check for POU (CIFAR10 instance-level unlearning). Each criterion threshold is calibrated on retained samples to achieve FPR = 1%. We report the achieved FPR and the TPR for detecting unlearned samples as outliers.

Criterion	**FPR on retained** ↓	**TPR on unlearned** ↑
3σ-fraction (ours)	0.98%	100.00%
Nearest-center L_2 (quantile)	1.00%	100.00%
Mahalanobis (Ledoit–Wolf)	1.00%	100.00%

Table 8: POU hyperparameter ablations (CIFAR-10 instance unlearning). Metrics are reported for the retained set unless noted: Remaining accuracy (Rem), Unlearned accuracy (Unl), Test accuracy (Test), Trustworthiness (TW, k=5), and mean L_2 drift. "—" indicates not logged for that run.

Group	$\boldsymbol{\lambda}_{\text{frozen}}$	$\boldsymbol{\lambda}_{\text{pull}}$	**Chunk**	**Rem** ↑	**Unl** ↓	**Test** ↑	**TW** ↑	$\boldsymbol{L_2}$ ↓
(A) Frozen-loss weight ablation (λ_{pull}=1, chunk= 100)								
	1.0	1.0	100	98.95%	0.00%	86.72%	0.9991	14.7758
	5.0	1.0	100	99.06%	0.00%	87.01%	1.0000	2.8754
	10.0	1.0	100	99.05%	0.00%	87.05%	1.0000	0.9672
	20.0	1.0	100	99.07%	0.00%	87.13%	1.0000	1.0185
	50.0	1.0	100	99.07%	0.00%	86.89%	1.0000	0.9451
(B) Pull-loss weight ablation (λ_{frozen}=20, chunk= 100)								
	20.0	0.25	100	99.06%	0.00%	87.11%	1.0000	1.0254
	20.0	0.5	100	99.07%	0.00%	86.97%	1.0000	0.9669
	20.0	2.0	100	99.08%	0.00%	87.08%	1.0000	1.2187
	20.0	5.0	100	99.06%	0.00%	87.03%	1.0000	3.3953
	20.0	10.0	100	98.91%	0.00%	86.46%	0.9996	10.1666
(C) Chunk size ablation (λ_{frozen}=20, λ_{pull}=10)								
	20.0	10.0	50	99.10%	0.00%	87.09%	1.0000	6.3323
	20.0	10.0	100	98.91%	0.00%	86.46%	0.9996	10.1666
	20.0	10.0	200	98.88%	0.00%	86.31%	0.9979	18.8360

Algorithm 1 Pseudocode for Pull-to-Outlier Unlearning (POU)

Input: Pretrained MLP head f_θ, training features $\mathbf{Z}$, forget indices $\mathcal{U}$, retained indices $\mathcal{N}$, margin δ, number of experts K
Initialize: Copy K expert MLPs from f_θ as $\{\theta_k\}_{k=1}^{K}$
Split $\mathcal{U}$ into K disjoint chunks: $\mathcal{U}_1, \ldots, \mathcal{U}_K$
for $k = 1$ **to** K **do**
  Compute global bounds $g_{\min}, g_{\max}$ from retained set $\mathcal{N}$
  For each $i \in \mathcal{U}_k$, compute outlier target τ_i using δ, $g_{\min}$, $g_{\max}$
  Optimize θ_k on:
    Pull loss $\mathcal{L}_{\text{pull}} = \sum_{i \in \mathcal{U}_k} \|f_{\theta_k}(z_i) - \tau_i\|^2$
    Frozen loss $\mathcal{L}_{\text{frozen}} = \sum_{j \in \mathcal{N}} \|f_{\theta_k}(z_j) - f_\theta(z_j)\|^2$
    Total loss: $\mathcal{L}_{\text{POU}} = \lambda_{\text{pull}} \cdot \mathcal{L}_{\text{pull}} + \lambda_{\text{frozen}} \cdot \mathcal{L}_{\text{frozen}}$
end for
Aggregation:
  For retained samples, average outputs from all experts
  For unlearned samples, select best expert based on closest match to τ_i
Evaluation: Run nearest-prototype classification and 3σ outlier check

Algorithm 2 Pseudocode for Contrastive Objective-Level Unlearning (COU)

Input: Pretrained projection f_θ, features $\mathbf{X}$, labels $\mathbf{y}$, forget set $\mathcal{U}$, prototypes $\{p_c\}$, epochs E
Stage 1: Learning Arbitrary Objective
for $e = 1$ **to** E **do**
  For each $i \in \mathcal{U}$:
    Compute second-nearest prototype $p_{c'}$, $c' \neq y_i$
    Apply perturbation loss:

$$\mathcal{L}_{\text{perturb}} = \sum_{i \in \mathcal{U}} \left[1 - \max_{c \neq y_i} \cos(f_\theta(x_i), p_c) \right]$$

  Update θ via $\mathcal{L}_{\text{perturb}}$
end for
Stage 2: Unlearning
For each $i \in \mathcal{U}$:
  Retrieve original embedding $z_i^{(0)}$
  Apply contrastive pull-back loss:

$$\mathcal{L}_{\text{restore}} = \sum_{i \in \mathcal{U}} \left[1 - \cos(f_\theta(x_i), z_i^{(0)}) \right]$$

Update θ via $\mathcal{L}_{\text{restore}}$
Evaluation: Measure trustworthiness and L_2 drift on full dataset before and after both stages