CAS2105 Homework 6: Mini AI Pipeline Project 🤗

Sunghyun Park (2020162029)

1 Introduction

This project explores the concept of **machine unlearning** in a controlled image classification setting. Concretely, we fine-tune a pre-trained Vision Transformer (ViT) on the CIFAR-10 dataset and then attempt to remove the model's knowledge about a specific class (*Airplane*) without retraining from scratch.

The motivating scenario is that a deployed model may be trained on data that later needs to be removed (e.g., due to privacy regulations, data poisoning, or user requests such as "the right to be forgotten"). Retraining from scratch is often computationally infeasible, especially for large models, so it is important to study approximate unlearning procedures: given an already trained model, can we efficiently modify it so that it behaves as if it had never seen a particular subset of the training data?

In this mini project, I frame the problem as follows: starting from an "infected" model trained on all CIFAR-10 classes, I *unlearn* the airplane class (D_{forget}), while preserving performance on the remaining classes (D_{retain}). I compare a naive gradient-ascent baseline with two more structured methods — retain-only fine-tuning and a two-stage unlearning procedure — and evaluate them using accuracy, predictive entropy, Jensen–Shannon distance (JSD) to a gold standard model, and t-SNE visualisation of representation space.

2 Task Definition

Task description. Let f_θ be a classifier with parameters θ that maps an input image x to a probability distribution over $K = 10$ CIFAR-10 classes. The training data is split into a *forget set* D_{forget} (all images of a target class, airplane) and a *retain set* D_{retain} (images from the remaining nine classes). Given an "infected" model f_{θ_0} trained on $D_{\text{forget}} \cup D_{\text{retain}}$, the goal is to obtain an unlearned model $f_{\theta'}$ that:

- performs poorly on the target class (i.e., it effectively "forgets" airplanes), and
- maintains as much performance as possible on the retain classes.

Motivation. Large-scale models are often trained once and reused in many downstream applications. When part of the data must be removed, retraining is impractical; instead, we want a lightweight *unlearning* procedure. This project serves as a toy example of how different unlearning strategies affect both the "forgotten" class and the remaining utility.

Input / Output. The input is a $3 \times 224 \times 224$ RGB image (CIFAR-10 resized to match ViT input size). The output is a 10-dimensional probability vector $p_\theta(y \mid x)$ over CIFAR-10 classes.

Success criteria. In this project I use three proxy criteria:

1. **Accuracy on the forget class:** Test accuracy on airplane images ($D^{\text{test}}_{\text{forget}}$) should drop close to random guessing.
2. **Accuracy on retain classes:** Test accuracy on non-airplane images ($D^{\text{test}}_{\text{retain}}$) should remain reasonably high.
3. **Distributional similarity to a gold standard:** on the forget test set, the predictive distribution of the unlearned model should be close (in JSD) to a model trained from scratch only on D_{retain}.

3 Methods

This section describes the overall pipeline and three unlearning strategies: a naive gradient-ascent baseline, retain-only fine-tuning, and the proposed two-stage unlearning method.

3.1 Overall Pipeline

The pipeline consists of the following stages:

1. **Pre-training / Fine-tuning.** Starting from a ViT-Tiny model pre-trained on ImageNet (`vit_tiny_patch16_224` from `timm`), I replace the classification head with a linear layer of size 10 and fine-tune it on the full CIFAR-10 training set for 3 epochs using SGD with momentum 0.9 and learning rate 10^{-4}. This yields the *Original* (infected) model f_{θ_0}.
2. **Gold standard model.** As an oracle reference, I train a second model from the same initialisation but only on D_{retain} (i.e., with all airplane images removed from both train and validation). This *Gold Standard* model approximates the behaviour we would like an ideal unlearned model to have.
3. **Unlearning stage.** Given the Original model, I apply different unlearning procedures using the forget and retain subsets of the training data, producing:
   - a naive Gradient Ascent model,
   - a Retain-only Fine-tuning model, and
   - a Two-Stage Unlearning model (ours).

All experiments are run with batch size 64, learning rate 10^{-4}, and unlearning for 5 epochs.

3.2 Naive Baseline: Gradient Ascent on the Forget Set

The simplest idea is to "destroy" the model's knowledge of the target class by *maximizing* the classification loss on forget samples.

Let D_{forget} contain labeled examples (x_f, y_f) of the target class. The standard cross-entropy loss is

$$\mathcal{L}_{\text{CE}}(\theta; x_f, y_f) = -\log p_\theta(y_f \mid x_f).$$

The naive unlearning objective maximizes this loss.

$$\max_\theta \; E_{(x_f,y_f)\sim D_{\text{forget}}}\Big[\mathcal{L}_{\text{CE}}(\theta; x_f, y_f)\Big].$$

In practice, this is implemented by *minimizing* the negative loss.

$$\min_\theta \; -E_{(x_f,y_f)\sim D_{\text{forget}}}\Big[\mathcal{L}_{\text{CE}}(\theta; x_f, y_f)\Big].$$

During training, I iterate only over the forget data loader and perform SGD updates with this modified loss.

3.3 Retain-only Fine-tuning

This strategy is to fine-tune the model only on retain data, hoping that the information about the target class will gradually fade as the decision boundaries are updated.

Let D_{retain} be the dataset consisting of all non-target images and labels. The fine-tuning objective is the standard cross-entropy:

$$\min_{\theta} \; E_{(x_r,y_r)\sim D_{\text{retain}}}\Big[\mathcal{L}_{\text{CE}}(\theta; x_r, y_r)\Big].$$

This is identical to usual supervised training, but the data loader only yields retain examples.

This method strongly preserves utility on D_{retain}, but there is no explicit pressure to remove knowledge about target images, so we expect it to under-unlearn the forget class.

3.4 Two-stage Unlearning (Ours)

The proposed method attempts to balance forgetting and retaining by decomposing unlearning into two stages.

Stage 1: Confusing the model on the forget class

In the first stage, I enforce that the model's predictions on airplane images become close to a uniform distribution over classes. Let u denote the uniform distribution over 10 classes. For each forget sample x_f, the target distribution is u, and I minimize the KL divergence:

$$\min_{\theta} \; E_{x_f\sim D_{\text{forget}}}\Big[D_{\text{KL}}\big(u \,\|\, p_\theta(\cdot \mid x_f)\big)\Big].$$

This is implemented as a single optimization step on one batch of forget data using `KLDivLoss` between $\log p_\theta$ and a uniform target.

Stage 2: Contrastive forgetting and retain healing

After Stage 1, the model might have lost some structure around airplane representations but can still be entangled with retain classes. Stage 2 jointly

1. pushes forget and retain logits apart via a contrastive term, and
2. preserves performance on the retain set via a standard classification term.

In each iteration, I sample a batch B_f from D_{forget} and a batch B_r from D_{retain}. Let $z_f = f_\theta(B_f)$ and $z_r = f_\theta(B_r)$ denote the logits.

I construct a similarity matrix $S = z_f z_r^\top/\tau$ (with $\tau = 1.15$) and define a contrastive loss that penalises high similarity between forget and retain logits:

$$L_{\text{contrastive}} = -\frac{1}{|B_f|}\sum_i \log\Big(\text{softmax}(S_{i,\cdot})_j\Big),$$

where I treat each forget logit as a query and all retain logits as "negatives". In practice, I compute

$$L_{\text{contrastive}} = \big(-\log \text{softmax}(S, \text{dim} = -1)\big)_{\text{mean}}$$

At the same time, I include a standard retain cross-entropy loss

$$L_{\text{retain}} = E_{(x_r,y_r)\in B_r}\left[\mathcal{L}_{\text{CE}}(\theta; x_r, y_r)\right].$$

The total Stage-2 loss is:

$$L_{\text{stage2}}(\theta) = L_{\text{contrastive}}(\theta) + L_{\text{retain}}(\theta).$$

By optimizing this objective for 5 epochs, the model is encouraged to: (1) stay useful on retain classes, and (2) shape its representation so that forget examples are pushed away from retain decision regions, yielding high entropy and low accuracy on the forget class.

4 Experiments

4.1 Dataset

I use the standard CIFAR-10 dataset, consisting of 50,000 training and 10,000 test images of resolution 32×32 across 10 classes. All experiments treat class index 0 (`"airplane"`) as the forget class.

- **Source:** CIFAR-10 from `torchvision.datasets`.
- **Total examples:** 50,000 train, 10,000 test.
- **Splits:** I use the standard training/test split. Within the training set, airplane images form D_{forget} (approx. 5,000 examples), and the remaining nine classes form D_{retain} (approx. 45,000 examples). The test set is similarly split into $D^{\text{test}}_{\text{forget}}$ and $D^{\text{test}}_{\text{retain}}$.
- **Preprocessing:** All images are resized to 224×224, converted to tensors, and normalised with mean and standard deviation $(0.5, 0.5, 0.5)$.

4.2 Metrics

To evaluate unlearning quality, I use four metrics:

- **Forget Accuracy (Train Forget):** Accuracy of the model on the training subset D_{forget}. A successful unlearning procedure should drive this value close to zero.
- **Retain Accuracy (Test Retain):** Accuracy on $D^{\text{test}}_{\text{retain}}$. This measures how much useful performance is preserved on non-forget classes.
- **Entropy on Forget Test:** the mean predictive entropy

$$H(p_\theta(\cdot \mid x)) = -\sum_y p_\theta(y \mid x) \log p_\theta(y \mid x)$$

  over $x \in D^{\text{test}}_{\text{forget}}$. If the model truly "does not know" how to classify airplane images, entropy should be close to $\log 10 \approx 2.30$, corresponding to a uniform prediction.
- **Jensen–Shannon distance to Gold:** For each $x \in D^{\text{test}}_{\text{forget}}$, I compute the Jensen–Shannon distance between the predictive distribution of the unlearned model and that of the Gold Standard model. Lower values mean that the unlearned model behaves similarly to a model that was never trained on the forget class.

4.3 Results

Quantitative results

Table 1 shows the metrics for five models: Original, Gold Standard, Fine-tuning, Gradient Ascent, and the proposed 2-Stage method.

The Original model performs strongly on both forget and retain data, as expected from a normally trained classifier. The Gold Standard, which never sees forget dataset, achieves high retain accuracy and essentially zero accuracy on the forget class, with moderate entropy on forget images.

Retain-only fine-tuning preserves retain accuracy (slightly improving it) but only weakly reduces forget performance (Forget Acc 78.28%), indicating that this approach under-unlearns the target class.

Table 1: Comparison of unlearning methods. "Forget Acc" is accuracy on the train forget set, and "Retain Acc" is accuracy on the test retain set. Entropy is measured on the forget test set, and JSD is the Jensen–Shannon distance to the Gold Standard model on the forget test set.

Method	**Forget Acc**	**Retain Acc**	**Entropy (F)**	**JSD (vs Gold)**
Original	98.94	96.08	0.0867	0.7961
Gold Standard	0.00	96.31	0.6868	0.0000
Fine-tuning	78.28	96.47	0.3097	0.6515
Gradient Ascent	0.00	11.11	0.0000	0.5565
2-Stage (Ours)	1.02	76.93	2.2814	0.5922

The naive Gradient Ascent method completely destroys forget performance but also destroys nearly all utility. Retain Acc drops to 11.11%, essentially random guessing. Moreover, entropy on the forget test set is almost zero.

The proposed 2-Stage method almost matches the Gold Standard in terms of forget accuracy (1.02%) and achieves near-maximal entropy (2.2814, close to $\log 10$), indicating high uncertainty on forget dataset. At the same time, it preserves a non-trivial retain accuracy of 76.93%, far higher than the Gradient Ascend baseline, though still below fine-tuning and Original models. JSD shows that 2-Stage is closer to the Gold Standard than the Original or Fine-tuning.

Representation analysis via t-SNE

To better understand how unlearning affects internal representations, I extract features from the `forward_features` layer of each model for a subset of retain and forget images and project them to 2D using t-SNE. Figure 1 shows the resulting scatter plots.

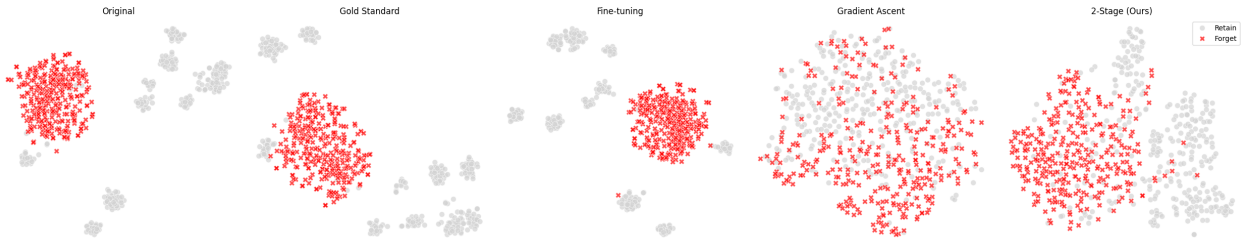

Figure 1: t-SNE visualisation of representations for retain (grey) and forget (red) samples across different models. From left to right: Original, Gold Standard, Fine-tuning, Gradient Ascent, 2-Stage (Ours).

In the Original model, forget samples form a tight, well-separated cluster, indicating that the model has a specialized representation for forget dataset. The Gold Standard, which never saw forget dataset, maps forget samples into more diffuse regions interleaved with retain clusters.

After retain-only fine-tuning, the airplane cluster is still clearly visible and compact, consistent with the high forget accuracy. the model still has a strong and distinct representation. Under Gradient Ascent, both forget and retain clusters collapse into a disorganized cloud.

Finally, the 2-Stage method leads to forget samples that are more dispersed and no longer form a tight, class-specific cluster, while retain samples still exhibit some structure. This aligns with the quantitative results: the model has largely removed a strong representation for airplanes while maintaining a usable representation for other classes.

5 Reflection and Limitations

Overall, this mini project demonstrates that naive unlearning strategies are either too weak or too destructive. Retain-only fine-tuning is attractive because it preserves accuracy on non-target classes, but it barely reduces performance on the forget class. Gradient Ascent on forget data successfully eliminates knowledge of the target class but also erases almost all useful information, making the model unusable in practice.

The proposed 2-Stage method strikes a more interesting balance. By first enforcing uniform predictions on the forget class and then jointly optimizing a contrastive loss and retain cross-entropy, the model achieves low forget accuracy and high entropy on airplane images while preserving a reasonable level of utility. In t-SNE, the forget cluster disappears, but retain clusters remain partially intact.

However, the method is still far from perfect. There is a significant drop in retain accuracy compared to the Original and Gold models, and the choice of hyperparameters was only lightly tuned.

If I had more time, I would explore more principled regularization terms that explicitly constrain the distance to the Gold model.

In summary, this project gave me hands-on experience in formulating a concrete unlearning problem, implementing several baselines and an improved pipeline, and critically examining the trade-off between forgetting and utility using both quantitative metrics and representation visualization.