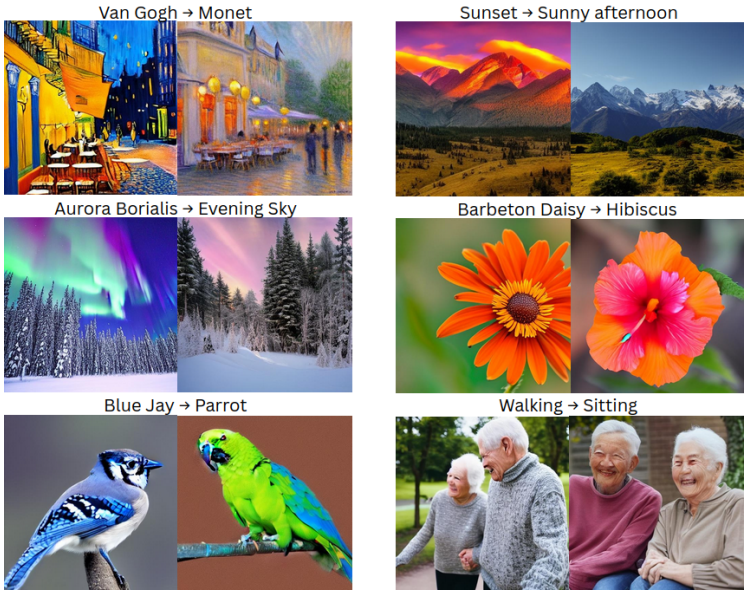
Editing Concepts in Diffusion Models

Om Prakash Choudhary **Saksham Agrawal**
Indian Institute of Science
{omprakashc, sakshama}@iisc.ac.in



Abstract

Diffusion models have recently emerged as a powerful paradigm for generating highly realistic and controllable images. While this enables many creative applications, it also opens the door to harmful content generation, raising serious ethical concerns around privacy, misuse, and safety. To address these risks, recent studies have explored fine-tuning and model editing techniques to *unlearn* harmful concepts without the expense of full retraining. However, these approaches often compromise the quality of unrelated generations. In this work, we introduce null-space constrained model editing to edit specific concepts while preserving the fidelity of unrelated concepts. Evaluated on **Stable Diffusion 1.4[1]**, our method demonstrates a $\mathbf{20}\%$ **improvement in forgetting** over leading baselines, establishing a more reliable pathway toward safer and responsible deployment of diffusion models. Our code is available at: GitHub.

1 Introduction

Recent advances in diffusion models have enabled the generation of highly realistic and controllable images at scale. However, these models are typically trained on massive datasets such as LAION-5B [2], which are known to contain harmful content, copyrighted material, and artistic styles. As a result, diffusion models can reproduce or even amplify this sensitive content, raising serious ethical concerns around privacy, misuse, and safety. These issues have motivated legal frameworks such as the General Data Protection Regulation (GDPR) and the California Consumer Privacy Act (CCPA),

which establish a "right to be forgotten", emphasizing urgency of developing effective methods to prevent harmful generation from diffusion models.

This has given rise to a new line of research on concept unlearning in diffusion models [3, 4, 5, 6]. Existing approaches primarily focus on fine-tuning or editing the UNet component to suppress higher-level concepts without requiring costly full retraining. A summary of these methods is provided in Section 2.

In this work, we investigate concept editing in the CLIP text encoder, guided by the UNet. Inspired by recent advances in null-space constrained editing for large language models (AlphaEdit [7]), we extend these ideas to the diffusion setting and explore their application to edit the CLIP encoder.

2 Prior Work

2.1 Data Unlearning

Data Unlearning [5] addresses the problem that diffusion models can memorize specific training examples. Formally, given a diffusion model ϵ_θ trained on dataset X with size n and given a forget set A with size k, the goal is to obtain a modified model $\epsilon_{\theta'}$ that behaves as if trained on data $X \setminus A$.

The unlearning objective is defined as

$$L_{X\setminus A}(\theta) = \sum_{x\in X\setminus A} \frac{1}{n-k} \mathbb{E}_{\substack{t\sim\{1,\dots,T\}\\ x_t\sim q(x_t|x)}} \|\epsilon - \epsilon_\theta(x_t, t)\|^2. \tag{1}$$

Reformulation. The authors show that Eq. (1) can be equivalently expressed as:

Proposition 1.

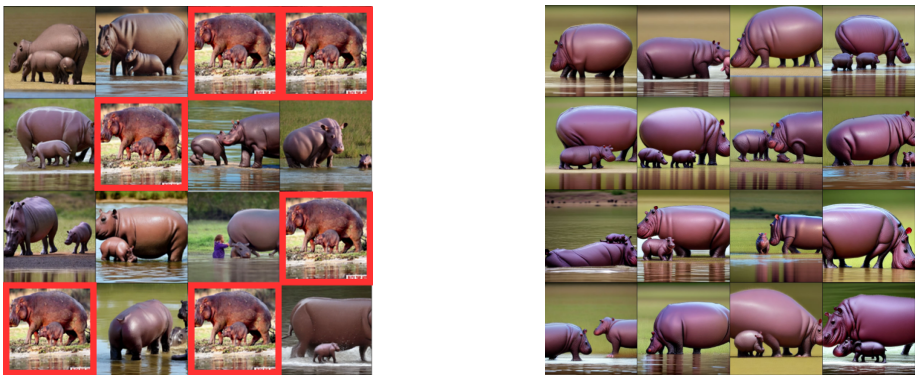
$$L_{X\setminus A}(\theta) = \frac{n}{n-k} \mathbb{E}_{x\in X} \mathbb{E}_{\substack{t\sim\{1,\cdots,T\}\\ x_t\sim q(x_t|x)}} \|\epsilon - \epsilon_\theta(x_t, t)\|^2 - \frac{k}{n-k} \mathbb{E}_{a\in A} \mathbb{E}_{\substack{t\sim\{1,\cdots,T\}\\ a_t\sim q(a_t|a)}} \|\epsilon - \epsilon_\theta(a_t, t)\|^2.$$

While mathematically correct, this formulation requires two separate passes of the diffusion model ϵ_θ, one over X and another over A. To reduce computational cost, the authors introduce a weighted importance sampling scheme:

$$q_\lambda(m_t \mid x, a) = (1-\lambda) q(m_t \mid x) + \lambda q(m_t \mid a),$$

where λ is a weighting parameter. This allows computing the loss in a **single model pass**, with experiments showing performance comparable to the naive two-pass method.

Results. The proposed method demonstrates robust forgetting of specific data points while preserving generalization to the rest of the dataset. It outperforms existing baselines such as **NEG-GRAD** and **ERASE-DIFF** in terms of both efficiency and effectiveness.



(a) **Before Unlearning**: 6 memorized samples (marked in red).

(b) **After Unlearning**: 0 memorized samples.

Figure 1: Qualitative results of the unlearning method from [5].

2.2 Erasing Concepts in Stable Diffusion (ESD)

ESD [3] addresses the problem of unlearning concepts by directly modifying the image distribution learned by the diffusion model ϵ_θ. The method leverages classifier-free guidance (CFG), where given a concept c, samples are drawn from

$$P_\theta(x|c) \propto P_\theta(x)\big(P_\theta(c|x)\big)^\eta,$$

with η denoting the guidance scale. Intuitively, larger η sharpens the distribution towards c, producing samples that more strongly resemble the concept.

To erase c, ESD inverts the CFG weighting by replacing η with $-\eta$. Fine-tuning on these modified samples leads to which has the effect of increasing density of other concepts which has effect of unlearning c (See Appendix Appendix A.2)

Proposition 2.

$$P_{\theta'}(x) \propto P_\theta(x)\, P_\theta(c|x)^{-\eta}.$$

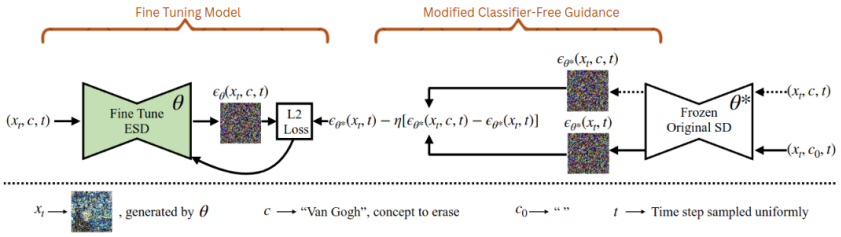


Figure 2: Overview of ESD [3].

See section Section 5 for quantitative evaluation of ESD. This algorithm serves as a strong baseline for evaluating unlearning algorithms.

2.3 Unified Concept Editing (UCE)

UCE [4] addresses the problem of *scalable concept editing*, enabling simultaneous modification of multiple concepts with extremely fast runtime (<1 sec). The method builds on the idea that cross-attention key and value weights encode implicit concept knowledge [8], and thus provides a handle for editing.

Formally, let (E, E^*) denote the set of edit prompts and corresponding target prompts, and let P be the set of preserve prompts. Consider a linear layer $W \in \mathbb{R}^{d_1 \times d_0}$ (typically $d_1 = d_0$, ranging from 1000–5000) with input text embedding c_i. UCE formulates a multi-objective optimization problemProposition 3:

Proposition 3 (UCE Objective)**.**

$$\min_W \sum_{\substack{c_i \in E \\ c_i^* \sim E^*}} \|Wc_i - W^{OLD}c_i^*\|_2^2 \; + \sum_{c_i \in P} \|Wc_i - W^{OLD}c_i\|_2^2,$$

where the first term enforces editing, and the second term preserves concepts.

Fortunately, optimization problem Proposition 3 admits a **closed-form solution**:

$$W = \left(\sum_{\substack{c_i \in E \\ c_i^* \in E^*}} W^{\text{OLD}} c_i^* c_i^T + \sum_{c_j \in P} W^{\text{OLD}} c_j c_j^T \right) \left(\sum_{c_i \in E} c_i c_i^T + \sum_{c_j \in P} c_j c_j^T \right)^{-1}.$$

For completeness, the derivation of this solution is provided in Appendix **??**.

Table 2 reports the performance of UCE. While not always the most accurate method, its has remarkable speed (<1 sec).

2.4 Forget Me Not

Forget-Me-Not is a plug-and-play, efficient and effective concept forgetting and correction method for large-scale text-to-image models. It provides an efficient way to forget specific concepts with as few as 35 optimization steps, which typically takes about 30 seconds.

Methodology

FMN works on a concept called "Attention Resteering". The context embeddings of the forgetting concept is located and the attention maps between these embeddings and the input features are computed (input feature is the noisy image in this context).
The objective is to minimize the attention maps and backpropagate the network. This method essentially fine-tunes U-Net to minimize each of the intermediate attention maps associated with the target concepts to forget. The algorithm for FMN is given below.

Algorithm 1 Forget-Me-Not on diffuser

Require: Context embeddings $\mathcal{C}$ containing the forgetting concept, embedding locations $\mathcal{N}$ of the forgetting concept, reference images $\mathcal{R}$ of the forgetting concept, diffuser G_θ, diffusion step T.

repeat
  $t \sim \text{Uniform}([1 \dots T]); \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  $r_i \sim \mathcal{R}; c_j, n_j \sim \mathcal{C}, \mathcal{N}$
  $x_0 \leftarrow r_i$
  $x_t \leftarrow \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon$
  ▷ $\bar{\alpha}_t$: noise variance schedule
  $x'_{t-1}, A_t \leftarrow G_\theta(x_t, c_j, t)$
  ▷ A_t: all attention maps
  $\mathcal{L} \leftarrow \sum_{a_t \in A_t} \|a_t^{[nj]}\|^2$
  ▷ $\mathcal{L}$: attention resteering loss
  $\theta \leftarrow \theta - \nabla_\theta \mathcal{L}$
until Concept forgotten

Figure 3: FMN Algorithm [9]

2.5 Concept Ablation (CA)

CA[6] aims at modifying the conditional distribution of the model given a target concept $p_\phi(\mathbf{x}|\mathbf{c}^*)$ to match a distribution $p(\mathbf{x}|\mathbf{c})$ defined by the anchor concept $\mathbf{c}$. This is done by minimizing KL Divergence between the two conditional distributions. There are two proposed methods in the paper to achieve the desired results:

1. Fine-tuning the model to match the model prediction between two text prompts containing the target and corresponding anchor concepts.
2. Defining conditional distribution $p(\mathbf{x}|\mathbf{c})$ by the modified text-image pairs of: a target concept prompt, paired with images of anchor concepts.

Formulation

The method assumes that the user provides the anchor concept (for example, if the forgetting concept is `A black dog`, the anchor concept can be something similar to `A Dog`). Thus, given a set of prompts $\{\mathbf{c}^*\}$ describing the target concept, the method aims to minimize the KL divergence between

$$\mathcal{D}_{\text{KL}}\big(p(\mathbf{x}_{0\dots T} \mid \mathbf{c}) \parallel p_{\hat{\Phi}}(\mathbf{x}_{0\dots T} \mid \mathbf{c}^*)\big) \tag{2}$$

Model Based Concept Ablation: Here, the pretrained distribution on the anchor concept is matched with the distribution of the target concept. Equation (2) can also be formulated as

$$\arg\min_{\hat{\Phi}} \sum_{t=1}^{T} \mathbb{E}_{p_\Phi(x_0 \dots x_T|c)} \left[\log \frac{p_\Phi(x_{t-1} \mid x_t, \mathbf{c})}{p_{\hat{\Phi}}(x_{t-1} \mid x_t, \mathbf{c}^*)} \right] \tag{3}$$

where Φ is the original model and $\hat{\Phi}$ is the new model we want to learn.

Proposition 4. *Optimizing* (3) *is equivalent to minimizing the following objective function:*

$$\arg\min_{\hat{\Phi}} \mathbb{E}_{\epsilon,\mathbf{x}_t,\mathbf{c}^*,\mathbf{c},t}\left[w_t \left\|\Phi(\mathbf{x}_t,\mathbf{c},t) - \hat{\Phi}(\mathbf{x}_t,\mathbf{c}^*,t)\right\|\right] \tag{4}$$

Noise Based Concept Ablation: Alternatively, we can redefine the ground truth text-image pairs as `<target concept text prompt, the generated image of the corresponding anchor concept text prompt>`. The diffusion model then can be fine-tuned on these pairs using the standard diffusion training loss.

$$\mathcal{L}_{\text{noise}}(\mathbf{x},\mathbf{c},\mathbf{c}^*) = \mathbb{E}_{\epsilon,\mathbf{x},\mathbf{c}^*,t}\left[w_t \left\|\epsilon - \hat{\Phi}(\mathbf{x}_t,\mathbf{c}^*,t)\right\|\right] \tag{5}$$

Comments on Concept Ablation:

1. The Model Based Concept Ablation can be done in two ways. Keeping two large networks Φ and $\hat{\Phi}$ would make the process computationally inefficient. The authors then assume that the conditional distribution of the anchor concept does not change in $\hat{\Phi}$ which is a big assumption as the target and anchor concept are still very closely related.
2. The model would be ineffective over indirect and adversarial prompts as the distribution over those prompts would still be the same. The authors mention this drawback in their Limitations section.

3 Preliminaries

Stable Diffusion consists of three main components: (i) a UNet that operates in the latent space $(4, 64, 64)$, (ii) a decoder that maps the latent representation to an RGB image $(3, 512, 512)$, and (iii) a CLIP text encoder, which is crucial for classifier-free guidance. In this work, we focus on editing the CLIP text encoder.

Formally, let E denote the set of prompts containing the concept to be edited, E^* be set of prompts with the corresponding target concept, and P the set of prompts whose concepts should be preserved. Consider a linear layer $W \in \mathbb{R}^{d_1 \times d_0}$.

- Let $K_0 \in \mathbb{R}^{d_0 \times n}$ be the inputs of W corresponding to prompts in P, and define $V_0 := WK_0$, the outputs for the preservation set.
- Let $K_1 \in \mathbb{R}^{d_0 \times n'}$ be the inputs of W corresponding to prompts in E. We define V_1 as the outputs that generate similar image in UNet as the output of target prompts E^*. One natural choice is $V_1 := WK_1^*$, where K_1^* are the embeddings of E^*. However obtain V_1 guided by UNet to produce similar image as $V_1^* := WK_1^*$, a detailed discussion related to this choice and it's implementation is provided in Section 5.

With these definitions, our editing objective for a linear layer is given by:

$$\min_{\tilde{\Delta}} \left(\|(W+\tilde{\Delta})K_1 - V_1\|^2 \; + \; \|(W+\tilde{\Delta})K_0 - V_0\|^2\right) \tag{6}$$

where the first term enforces editing of the target concepts in E, and the second term enforces preservation of the concepts in P.

Although in principle any layer can be edited, prior work [10] suggests that factual knowledge is primarily stored in the feed-forward (FFN) layers following attention. This makes them a natural target for editing in our method.

4 Proposed Method

The objective in Eq. (6) admits a closed-form solution:

$$\tilde{\Delta} = (V_1 - WK_1)K_1^T\left(K_0K_0^T + K_1K_1^T\right)^{-1},$$

(refer Appendix C). However, prior work [7] has shown that this update can interfere with existing knowledge, i.e., it changes the outputs for K_0 (the preservation set).

To address this, we constrain the update to lie in the left null space of K_0. Formally, given $K_0 \in \mathbb{R}^{d_0 \times n}$, let $U \in \mathbb{R}^{d_0 \times n'}$ be its left null space (i.e. $UK_0 = 0$). Define the projection matrix:

$$P := UU^T \in \mathbb{R}^{d_0 \times d_0}.$$

This choice ensures $PK_0 = 0$, i.e., the preservation set remains unchanged.

Substituting P into the objective yields:

$$\begin{aligned}
&\min_{\Delta} \|(W + \Delta P)K_1 - V_1\|^2 + \|(W + \Delta P)K_0 - V_0\|^2 \\
&= \|(W + \Delta P)K_1 - V_1\|^2 + \|(WK_0 - V_0) + \Delta(PK_0)\|^2 \\
&= \|(W + \Delta P)K_1 - V_1\|^2 + \|0 + 0\|^2 \\
&= \|(W + \Delta P)K_1 - V_1\|^2. \qquad (7)
\end{aligned}$$

Thus, the null-space projection removes the preservation term entirely, guaranteeing that editing does not affect previously learned knowledge.

The simplified objective Eq. (7) also admits a closed-form solution (refer Appendix C):

$$\Delta_{\text{AlphaEdit}} = RK_1^T P \left(K_1 K_1^T P + I\right)^{-1},$$

where

$$R = V_1 - WK_1.$$

This formulation ensures that the update modifies the model to incorporate new knowledge (editing $E \mapsto E^*$) while leaving the preservation set P completely unaffected.

5 Experiments

All experiments are performed on Stable Diffusion 1.4 [1]. For evaluation, we prepared a dataset of 20 diverse concepts (including actions, art styles, and objects) with 10 prompts each (5 prompts are direct and another 5 are indirect, i.e. they refer to concept without mentioning them in prompt) using an LLM.

To assess unlearning, after unlearning a concept, we generated images of all concepts (20 images perconcept for 20 concepts so total 400 images per edited concept) and employed a **VLM LLava v1.5 7B**[11] to check for the concept in each image (for the unlearned concept, the images **should not** contain that concept, while for other concepts, they must be present in the image). The response of LLava is then used to compute the **forget and retention scores** for each concept.

In our method, editing a concept requires pairing it with a target concept. To this end, we used an LLM to generate 15 edit prompts for each concept along with 15 corresponding target prompts. The target concepts were chosen to be **semantically similar** to the edit concepts, and the edit–target prompt pairs were designed to share as much **contextual background** as possible, differing only in the target concept. Additionally, **edit prompts** from **other concepts** served as the **retention set** during editing of a given concept.

UNet-Guided Computation of V_1

As discussed in Section 3, a natural choice for the target representation is

$$V_1^* := WK_1^*,$$

where K_1^* are the inputs corresponding to the edit prompts E^*. However, since the number of edit prompts E^* is relatively small, directly using V_1^* often leads to overfitting.

Instead, we adopt a UNet-guided refinement strategy. We initialize $V_1^{(0)} := WK_1$ and iteratively refine it using gradient descent to align denoised images of edit and target prompts.

Algorithm 1 UNet-Guided Refinement of V_1

Require: Edit prompts E, target prompts E^*, layer W, learning rate η, steps T

for $t = 0$ to $T - 1$ **do**

  $I_{\text{orig}} \leftarrow \text{UNet}(V_1^{(t)}), \quad I_{\text{edit}} \leftarrow \text{UNet}(V_1^*)$

  $\mathcal{L} \leftarrow \text{MSE}(I_{\text{orig}}, I_{\text{edit}})$

  $V_1^{(t+1)} \leftarrow V_1^{(t)} - \eta \nabla_{V_1} \mathcal{L}$

end for

return $V_1^{(T)}$

Results:

Table 1: Unedited Stable Diffusion 1.4

Concept	Forget	Retain
AppleFruit	0.150	0.805
AuroraBorialis	0.150	0.808
BarbetonDaisy	0.250	0.813
BlueJay	0.100	0.808
Dancing	0.100	0.779
Doodle	0.050	0.797
Eating	0.450	0.811
GolfBall	0.150	0.805
Jumping	0.200	0.795
LabradorRetriever	0.200	0.811
Monet	0.750	0.850
Neon	0.000	0.784
Rainfall	0.050	0.805
Scenery	0.000	0.800
Sketch	0.000	0.784
Sleeping	0.450	0.818
Sunset	0.000	0.782
VanGogh	0.700	0.826
Walking	0.150	0.805
Wedding	0.000	0.774
Average	**0.195**	**0.803**

Table 2: Unified Concept Editing [4]

Concept	Forget	Retain
AppleFruit	0.300	0.795
AuroraBorialis	0.100	0.792
BarbetonDaisy	0.250	0.789
BlueJay	0.500	0.797
Dancing	0.150	0.782
Doodle	0.000	0.803
Eating	0.500	0.797
GolfBall	0.600	0.803
Jumping	0.200	0.795
LabradorRetriever	0.350	0.803
Monet	0.950	0.832
Neon	0.050	0.787
Rainfall	0.350	0.795
Scenery	0.000	0.776
Sketch	0.000	0.739
Sleeping	0.800	0.821
Sunset	0.500	0.755
VanGogh	1.000	0.853
Walking	0.200	0.784
Wedding	0.100	0.782
Average	**0.345**	**0.794**

Table 3: Erasing Stable Diffusion [3]

Concept	Forget	Retain
AppleFruit	0.375	0.802
AuroraBorialis	0.333	0.823
BarbetonDaisy	0.375	0.805
BlueJay	0.458	0.775
Dancing	0.292	0.825
Doodle	0.000	0.793
Eating	0.417	0.814
GolfBall	0.667	0.800
Jumping	0.208	0.823
LabradorRetriever	0.250	0.795
Monet	1.000	0.820
Neon	0.167	0.784
Rainfall	0.417	0.816
Scenery	0.000	0.800
Sketch	0.000	0.809
Sleeping	0.750	0.831
Sunset	0.375	0.818
VanGogh	0.950	0.818
Walking	0.208	0.798
Wedding	0.100	0.793
Average	**0.367**	**0.807**

Table 4: Our Method

Concept	Forget	Retain
AppleFruit	0.950	0.771
AuroraBorialis	0.200	0.768
BarbetonDaisy	0.450	0.705
BlueJay	0.600	0.755
Dancing	0.250	0.758
Doodle	0.150	0.671
Eating	0.850	0.792
GolfBall	0.850	0.813
Jumping	0.350	0.813
LabradorRetriever	1.000	0.792
Monet	1.000	0.713
Neon	0.550	0.789
Rainfall	0.600	0.771
Scenery	0.000	0.782
Sketch	0.050	0.803
Sleeping	0.600	0.779
Sunset	0.350	0.763
VanGough	0.900	0.816
Walking	0.400	0.795
Wedding	0.950	0.711
Average	**0.553**	**0.768**

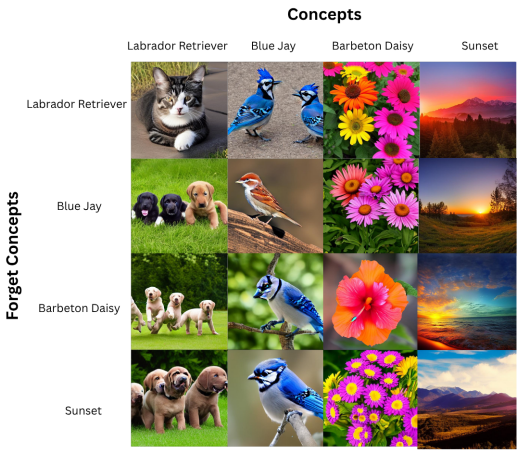


Figure 4: Qualitative Result of Single Concept Unlearning

Table 5: Harmonic Mean of Forget and Retain Scores

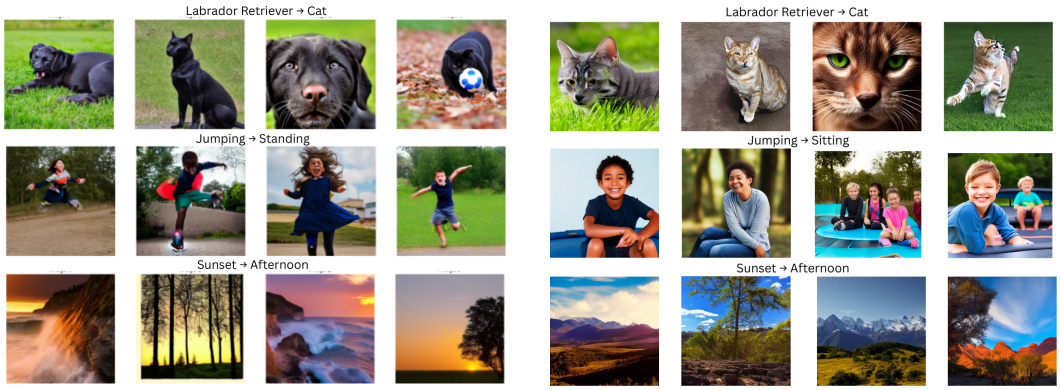
	Unedited	**UCE**	**ESD**	**Ours**
Harmonic Mean	0.313	0.480	0.504	0.642

Ablation Study

In principle, our method can be applied to any linear layer. However recent studies suggest modify the FFN layers for knowledge updates [10]. In our experiments, we edited all but the first FFN layer in attention blocks of the CLIP text encoder (a total of 11 layers). For each concept, we performed 10 edits to each layer, although we observed that reasonable results could be obtained with as few as a single edit.

We also experimented with editing FFN present in cross-attention block in the U-Net. For editing, our method projects the update onto the null space of K_0 (the inputs of the retention prompts to the given layer). However, in case of U-Net we found that the eigenvalues of $K_0 K_0^T$ were significantly larger (relative to their mean) compared to the CLIP layers, making it difficult to construct a reasonable null space.

Even without projecting updates (MEMIT), editing in the U-Net was much less efficient: it required at least 30–40 edits (compared to just 1 edit in the CLIP encoder) with worse quality, and for some concepts, it failed entirely regardless of the number of edits. We hypothesize that this inefficiency arises because the CLIP encoder directly encodes concept information, whereas the U-Net does not. Instead, the U-Net relies on cross-attention to incorporate concept embeddings from CLIP into the image generation process.



(a) Results of Editing U-Net after 40 edits

(b) Results of Editing CLIP Encoder after 10 edits

Figure 5: Comparison of editing performance: (a) U-Net vs. (b) CLIP Encoder.

6 Future Work

Scalability How far does this method scale without breaking down the model? That is, roughly how many concepts can we edit while maintaining the generation quality of unedited concepts.

Overfitting We notice that the model is prone to overfitting, especially since we have a limited number of handcrafted prompts. However, we also observe that gibberish-looking inputs can produce specific outputs (e.g., adversarial prompts producing "Blue Jay"). This opens the possibility of using random prompts and somehow injecting the subject to generate the image of that subject. This might remedy overfitting and we won't need handcrafted prompts for editing.

Which layers to edit? Methods like AlphaEdit and MEMIT are proposed for situations where we have a subject–object pair (e.g., "Eiffel Tower" $\rightarrow$ "Paris"). To find which layer should be edited (to change "Paris"), causal analysis is used.

In short, they take a sentence **unrelated** to the subject (e.g., "Taj Mahal is located in"), copy the activations of the original subject ("Eiffel Tower is located in") in each layer one at a time, run the later layers, and check which layer's output triggers "Paris".

However, our problem is different: we have an original and an edit concept pair, and it is unclear how to apply causal analysis here.

Also in LLMs, only the last token is used to predict the next token (in the example above, "is" will only be used to predict "Paris"), so causal analysis only needs to monitor the last token. In our case, the entire embedding will be used to generate the image.

Posterior Density Is it even possible to construct density $p'(x)$ (shown in Eq. (8))?

References

[1] Robin Rombach et al. "High-Resolution Image Synthesis With Latent Diffusion Models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 10684–10695 (cit. on pp. 1, 6).

[2] Christoph Schuhmann et al. *LAION-5B: An open large-scale dataset for training next generation image-text models*. 2022. arXiv: 2210.08402 [cs.CV]. URL: https://arxiv.org/abs/2210.08402 (cit. on p. 1).

[3] Rohit Gandikota et al. *Erasing Concepts from Diffusion Models*. 2023. arXiv: 2303.07345 [cs.CV]. URL: https://arxiv.org/abs/2303.07345 (cit. on pp. 2, 3, 7).

[4] Rohit Gandikota et al. *Unified Concept Editing in Diffusion Models*. 2024. arXiv: 2308.14761 [cs.CV]. URL: https://arxiv.org/abs/2308.14761 (cit. on pp. 2, 3, 7).

[5] Silas Alberti et al. *Data Unlearning in Diffusion Models*. 2025. arXiv: 2503.01034 [cs.LG]. URL: https://arxiv.org/abs/2503.01034 (cit. on p. 2).

[6] Nupur Kumari et al. *Ablating Concepts in Text-to-Image Diffusion Models*. 2023. arXiv: 2303.13516 [cs.CV]. URL: https://arxiv.org/abs/2303.13516 (cit. on pp. 2, 4).

[7] Junfeng Fang et al. *AlphaEdit: Null-Space Constrained Knowledge Editing for Language Models*. 2025. arXiv: 2410.02355 [cs.CL]. URL: https://arxiv.org/abs/2410.02355 (cit. on pp. 2, 5).

[8] Hadas Orgad, Bahjat Kawar, and Yonatan Belinkov. *Editing Implicit Assumptions in Text-to-Image Diffusion Models*. 2023. arXiv: 2303.08084 [cs.CV]. URL: https://arxiv.org/abs/2303.08084 (cit. on p. 3).

[9] Eric Zhang et al. *Forget-Me-Not: Learning to Forget in Text-to-Image Diffusion Models*. 2023. arXiv: 2303.17591 [cs.CV]. URL: https://arxiv.org/abs/2303.17591 (cit. on p. 4).

[10] Peter Hase et al. *Does Localization Inform Editing? Surprising Differences in Causality-Based Localization vs. Knowledge Editing in Language Models*. 2023. arXiv: 2301.04213 [cs.LG]. URL: https://arxiv.org/abs/2301.04213 (cit. on pp. 5, 8).

[11] Haotian Liu et al. *Improved Baselines with Visual Instruction Tuning*. 2023 (cit. on p. 6).

A Proofs

A.1 Proof of Proposition 1

Proof. We restate the proposition for convinience.

Proposition 5.

$$L_{X\setminus A}(\theta) = \frac{n}{n-k}\sum_{x\in X}\mathbb{E}_{\substack{t\sim\{1,\cdots,T\}\\ x_t\sim q(x_t|x)}}\|\epsilon-\epsilon_\theta(x_t,t)\|^2 - \frac{k}{n-k}\sum_{a\in A}\mathbb{E}_{\substack{t\sim\{1,\cdots,T\}\\ a_t\sim q(a_t|a)}}\|\epsilon-\epsilon_\theta(a_t,t)\|^2.$$

$$\begin{aligned}
L_{X\setminus A}(\theta) &= \sum_{x\in X\setminus A}\frac{1}{n-k}\mathbb{E}_{\substack{t\sim\{1,\ldots,T\}\\ x_t\sim q(x_t|x)}}\|\epsilon-\epsilon_\theta(x_t,t)\|^2.\\
&= \sum_{x\in X}\frac{1}{n-k}\mathbb{E}_{\substack{t\sim\{1,\cdots,T\}\\ x_t\sim q(x_t|x)}}\|\epsilon-\epsilon_\theta(x_t,t)\|^2 - \sum_{a\in A}\frac{1}{n-k}\mathbb{E}_{\substack{t\sim\{1,\cdots,T\}\\ a_t\sim q(a_t|a)}}\|\epsilon-\epsilon_\theta(a_t,t)\|^2.\\
&= \frac{n}{n-k}\mathbb{E}_{x\in X}\mathbb{E}_{\substack{t\sim\{1,\cdots,T\}\\ x_t\sim q(x_t|x)}}\|\epsilon-\epsilon_\theta(x_t,t)\|^2 - \frac{k}{n-k}\mathbb{E}_{a\in A}\mathbb{E}_{\substack{t\sim\{1,\cdots,T\}\\ a_t\sim q(a_t|a)}}\|\epsilon-\epsilon_\theta(a_t,t)\|^2.
\end{aligned}$$

□

A.2 Proof of Proposition 2

We illustrate the effect of ESD using a simple toy mixture model:

$$p(x) = \alpha_1\,\mathcal{N}(\mu_1,1) + \alpha_2\,\mathcal{N}(\mu_2,1) + \alpha_3\,\mathcal{N}(\mu_3,1),$$

where the latent variable $z\in\{1,2,3\}$ selects a component.

Ideal unlearning. Suppose we wish to remove class 3. The *ideal* unlearning distribution would simply reweight the remaining components:

$$p'(x) = \frac{\alpha_1}{\alpha_1+\alpha_2}\,\mathcal{N}(\mu_1,1) + \frac{\alpha_2}{\alpha_1+\alpha_2}\,\mathcal{N}(\mu_2,1). \tag{8}$$

This exactly preserves the relative mixture proportions of the surviving classes.

ESD update. In contrast, ESD modifies the density via

$$\tilde{p}(x) \;\propto\; \frac{p(x)}{p(z=3\mid x)^\eta},$$

where

$$p(z=3\mid x) = \frac{\alpha_3\,\mathcal{N}(\mu_3,1)}{\sum_{j=1}^3\alpha_j\,\mathcal{N}(\mu_j,1)}.$$

Expanding, we obtain

$$\tilde{p}(x) \;\propto\; \left(\sum_{j=1}^3\alpha_j\,\mathcal{N}(\mu_j,1)\right)^{\eta+1}\cdot\big(\alpha_3\,\mathcal{N}(\mu_3,1)\big)^{-\eta}.$$

Interpretation. While $\tilde{p}(x)$ is not equal to the ideal distribution (8), it has a useful property: the corrective factor

$$\big(\alpha_3\,\mathcal{N}(\mu_3,1)\big)^{-\eta}$$

penalizes regions of high density under class 3. Consequently, samples likely to originate from $z=3$ are strongly downweighted, while regions belonging to other classes are relatively amplified. This mechanism explains how ESD effectively reduces the model's ability to generate class 3, even though it does not perfectly match the ideal unlearning objective.

A.3 Proof of Proposition 4

Proof.

$$\mathcal{D}_{KL}(p_\theta(x_{0\dots T}|c) \parallel p_\phi(x_{0\dots T}|c^*)) = \mathbb{E}_{p_\theta(x_{0\dots T})} \log \frac{\prod_{t=1}^{T} p_\theta(x_{t-1}|x_t, c) p_\theta(x_T)}{\prod_{t=1}^{T} p_\phi(x_{t-1}|x_t, c^*) p_\phi(x_T)} \tag{8}$$

$$= \sum_{t=1}^{T} \mathbb{E}_{p_\theta(x_{0\dots T})} \log \frac{p_\theta(x_{t-1}|x_t, c)}{p_\phi(x_{t-1}|x_t, c^*)} \tag{9}$$

We expand the term corresponding to a particular time step $\hat{t}$, i.e.,

$$\mathbb{E}_{p_\theta(x_{0\dots T})} \log \frac{p_\theta(x_{\hat{t}-1}|x_{\hat{t}}, c)}{p_\phi(x_{\hat{t}-1}|x_{\hat{t}}, c^*)} = \int_{x(0\dots T)} \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t, c) p_\theta(x_T) \log \frac{p_\theta(x_{\hat{t}-1}|x_{\hat{t}}, c)}{p_\phi(x_{\hat{t}-1}|x_{\hat{t}}, c^*)} \, dx_{(0\dots T)} \tag{10}$$

$$= \int_{(x(\hat{t}\dots T)} p_\theta(x_{(\hat{t}\dots T)}|c) \left[\int \prod_{t=1}^{\hat{t}} p_\theta(x_{t-1}|x_t, c) \log \frac{p_\theta(x_{\hat{t}-1}|x_{\hat{t}}, c)}{p_\phi(x_{\hat{t}-1}|x_{\hat{t}}, c^*)} \, dx_{(t=0\dots\hat{t}-1)} \right] dx_{(\hat{t}\dots T)} \tag{11}$$

$$= \int p_\theta(x_{\hat{t}}|c) \left[\int \left(\prod_{t=1}^{\hat{t}} p_\theta(x_{t-1}|x_t, c) \right) \log \frac{p_\theta(x_{\hat{t}-1}|x_{\hat{t}}, c)}{p_\phi(x_{\hat{t}-1}|x_{\hat{t}}, c^*)} \, dx_{(t=1\dots\hat{t}-1)} \right] dx_{\hat{t}} \tag{12}$$

$$= \int p_\phi(x_{\hat{t}}|c) \left[\int_{x_{\hat{t}-1}} p_\phi(x_{\hat{t}-1}|x_{\hat{t}}, c) \log \frac{p_\theta(x_{\hat{t}-1}|x_{\hat{t}}, c)}{p_\phi(x_{\hat{t}-1}|x_{\hat{t}}, c^*)} \right. \tag{13}$$

$$\left. \times \left(\int_{x_{(0\dots\hat{t}-2)}} \prod_{t=1}^{\hat{t}-1} p_\phi(x_{t-1}|x_t, c) \, dx_{(\hat{t}-2\dots 0)} \right) dx_{\hat{t}-1} \right] dx_{\hat{t}} \tag{14}$$

The integral over $dx_{(\hat{t}-2\dots 0)}$ will be 1 since it is an integration of the probability distribution over the range it is defined. Thus the previous term can be re-written as,

$$\mathbb{E}_{x_{\hat{t}} \sim p_\phi(x_{\hat{t}}|c)} \left[\int_{x_{\hat{t}-1}} p_\phi(x_{\hat{t}-1}|x_{\hat{t}}, c) \log \frac{p_\theta(x_{\hat{t}-1}|x_{\hat{t}}, c)}{p_\phi(x_{\hat{t}-1}|x_{\hat{t}}, c^*)} \, dx_{\hat{t}-1} \right] \tag{15}$$

$$= \mathbb{E}_{x_{\hat{t}} \sim p_\phi(x_{\hat{t}}|c)} \Big[\mathcal{D}_{KL}(p_\phi(x_{\hat{t}-1}|x_{\hat{t}}, c) \parallel p_\phi(x_{\hat{t}-1}|x_{\hat{t}}, c^*)) \Big] \tag{16}$$

$$= \mathbb{E}_{x_{\hat{t}} \sim p_\phi(x_{\hat{t}}|c)} \Big[\eta \big(\Phi(x_{\hat{t}}, c, t) - \hat{\Phi}(x_{\hat{t}}, c^*, t) \big)^2 \Big] \tag{17}$$

□

B UCE

Proposition 6 (UCE Objective)**.**

$$\min_W \sum_{\substack{c_i \in E \\ c_i^* \sim E^*}} \|W c_i - W^{OLD} c_i^*\|_2^2 \; + \; \sum_{c_i \in P} \|W c_i - W^{OLD} c_i\|_2^2,$$

Proof. Define the following column-wise data matrices:

$$C_E := \big[c_i \big]_{c_i \in E} \in \mathbb{R}^{d_0 \times n_E}, \qquad C_{E^*} := \big[c_i^* \big]_{c_i^* \in E^*} \in \mathbb{R}^{d_0 \times n_E},$$

$$C_P := \big[\, c_j \big]_{c_j \in P} \in \mathbb{R}^{d_0 \times n_P},$$

and collect them into block matrices

$$C := \big[\, C_E \;,\; C_P \,\big] \in \mathbb{R}^{d_0 \times n}, \qquad C^* := \big[\, C_{E^*} \;,\; C_P \,\big] \in \mathbb{R}^{d_0 \times n},$$

where $n = n_E + n_P$. Also let W^{OLD} denote the fixed (old) linear layer.

With this notation the objective in the proposition can be written compactly as a Frobenius-norm least squares problem:

$$\mathcal{L}(W) = \sum_{c_i \in E} \|W c_i - W^{\mathrm{OLD}} c_i^*\|_2^2 \;+\; \sum_{c_j \in P} \|W c_j - W^{\mathrm{OLD}} c_j\|_2^2 = \|WC - W^{\mathrm{OLD}} C^*\|_F^2.$$

Expand the squared Frobenius norm and take the derivative w.r.t. W. Using $\partial \|A\|_F^2 / \partial W = 2(WCC^T - W^{\mathrm{OLD}} C^* C^T)$ (standard matrix calculus), we obtain the normal equation

$$2\big(WCC^T - W^{\mathrm{OLD}} C^* C^T\big) = 0 \quad \Longrightarrow \quad WCC^T = W^{\mathrm{OLD}} C^* C^T.$$

Assuming the matrix $CC^T \in \mathbb{R}^{d_0 \times d_0}$ is invertible (full rank), we can solve for W by right-multiplying by $(CC^T)^{-1}$:

$$W = W^{\mathrm{OLD}} C^* C^T (CC^T)^{-1}.$$

Writing the products C^*C^T and CC^T as sums over columns recovers the componentwise form:

$$C^*C^T = \sum_{c_i \in E} c_i^* c_i^T \;+\; \sum_{c_j \in P} c_j c_j^T, \qquad CC^T = \sum_{c_i \in E} c_i c_i^T \;+\; \sum_{c_j \in P} c_j c_j^T.$$

Therefore the solution is equivalently written as

$$W = \left(\sum_{c_i \in E} W^{\mathrm{OLD}} c_i^* c_i^T \;+\; \sum_{c_j \in P} W^{\mathrm{OLD}} c_j c_j^T \right) \left(\sum_{c_i \in E} c_i c_i^T \;+\; \sum_{c_j \in P} c_j c_j^T \right)^{-1},$$

which is the stated closed form.

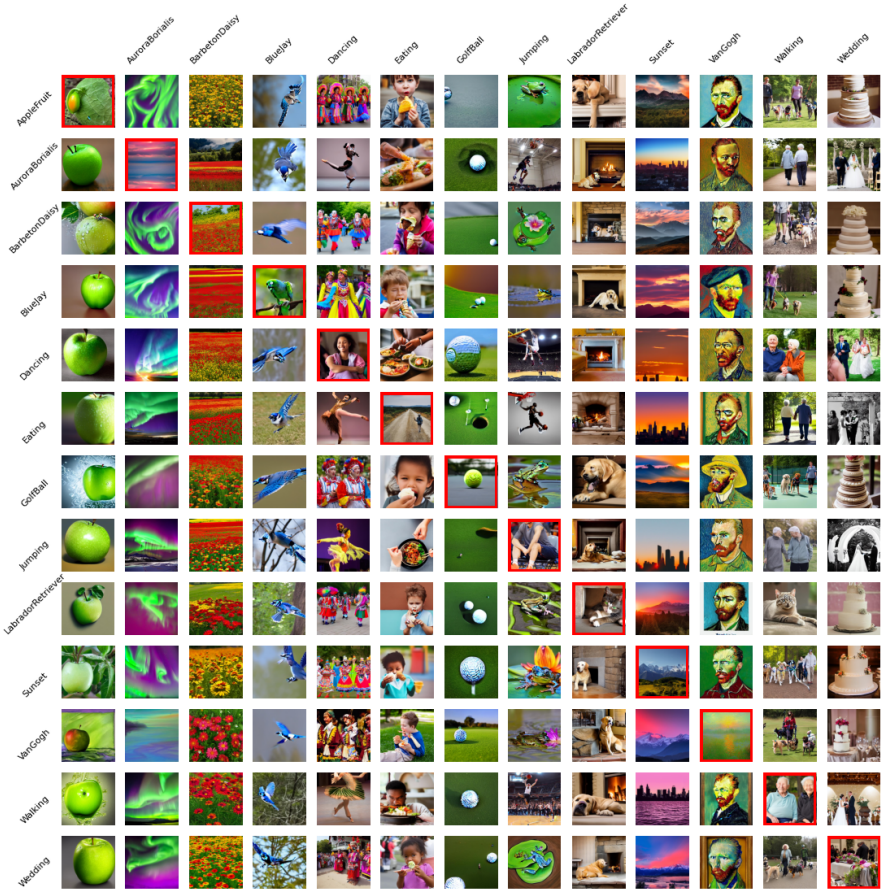
Rank caveat. If CC^T is singular (not full rank), the inverse above does not exist. A standard remedy is Tikhonov regularization (ridge):

$$W = W^{\mathrm{OLD}} C^* C^T \big(CC^T + \lambda I\big)^{-1},$$

for a small $\lambda > 0$, which both stabilizes the inversion and yields the unique minimum of the regularized objective $\|WC - W^{\mathrm{OLD}} C^*\|_F^2 + \lambda \|W\|_F^2$.

This completes the proof. □

C Our


AppleFruit
AuroraBorialis
BarbetonDaisy
BlueJay
Dancing
Eating
GolfBall
Jumping
LabradorRetriever
Sunset
VanGogh
Walking
Wedding