Embarrassingly Efficient Unlearning with SVD

Marcin Sendera [1] **Łukasz Struski** [1] **Kamil Ksiażek** [1] **Kryspin Musiol** [1 2] **Jacek Tabor** [1] **Dawid Rymarczyk** [1 3]

Abstract

While the capabilities of generative foundational models have advanced rapidly in recent years, methods to prevent harmful and unsafe behaviors remain underdeveloped. Among the pressing challenges in AI safety, machine unlearning (MU) has become increasingly critical to meet upcoming safety regulations. Most existing MU approaches focus on altering the most significant parameters of the model. However, these methods often require fine-tuning substantial portions of the model, resulting in high computational costs and training instabilities, which are typically mitigated by access to the original training dataset.

In this work, we address these limitations by leveraging Singular Value Decomposition (SVD) to create a compact, low-dimensional projection that enables the selective forgetting of specific data points. We propose Singular Value Decomposition for Efficient Machine Unlearning (SEMU), a novel approach designed to optimize MU in two key aspects. First, SEMU minimizes the number of model parameters that need to be modified, effectively removing unwanted knowledge while making only minimal changes to the model's weights. Second, SEMU eliminates the dependency on the original training dataset, preserving the model's previously acquired knowledge without additional data requirements.

1. Introduction

Machine unlearning is the process of modifying a model to ensure it does not memorize specific knowledge (Bourtoule et al., 2021). Depending on the context, this knowledge might involve harmful biases or private human data that must be removed for privacy reasons. However, implementing machine unlearning in deep neural networks is particularly challenging due to their tangled structure, where complex systems of neurons encode the learned information (Kurmanji et al., 2023).

[1]Faculty of Mathematics and Computer Science, Jagiellonian University [2]Institute of Theoretical and Applied Informatics, Polish Academy of Sciences [3]Ardigen SA. Correspondence to: Marcin Sendera <marcin.sendera@uj.edu.pl>.

ICML 2025 Workshop on Machine Unlearning for Generative AI

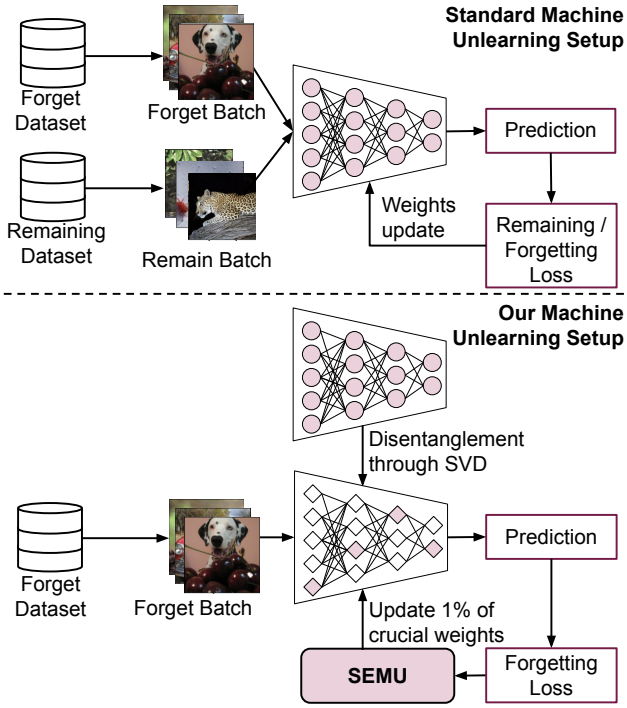

Figure 1. Illustration of the differences between the standard machine unlearning setup (top row) and our SEMU method (bottom row). Unlike the standard approach, SEMU does not need a remaining dataset, making it highly efficient in terms of data utilization. Moreover, SEMU modifies only a small fraction of the model's weights to remove specific knowledge. This sparsity is achieved through SVD projection (diamonds), which disentangles the weights and identifies the crucial ones for forget batch.

In typical machine unlearning setups, two datasets are used: the forget dataset, containing samples representing the knowledge to be removed, and the remaining dataset, which represents the knowledge to be retained (Kurmanji et al., 2023). The entire neural network is then adjusted to forget the unwanted information. However, this approach leads to two major inefficiencies. First, a large number of model parameters are altered (Sekhari et al., 2021), usually the entire model or a significant subset of its weights (Fan et al., 2024). Second, the process heavily depends on the remaining dataset (Jia et al., 2023; Kurmanji et al., 2023), which is used to preserve model's accuracy but requires additional computational costs.

To address these inefficiencies, we propose Singular Value Decomposition for Efficient Machine Unlearning (SEMU), a novel method leveraging SVD to identify a critical subset of model weights that need modification to forget specific data. By altering only a small subset of the model's weights, SEMU removes the need for the remaining dataset during the unlearning process. This makes SEMU not only efficient in terms of parameter alteration but also significantly reduces the iterations and data points required to retain knowledge. This efficiency is particularly valuable in scenarios where access to the training data during the unlearning is restricted due to privacy concerns.

SEMU achieves its improvements through a combination of gradient information and SVD. Gradients guide the model on how to adjust to satisfy unlearning loss, while SVD decomposes the model to pinpoint the weights critical to forgetting. As a result, SEMU minimizes the number of altered parameters and the usage of remaining dataset.

Through extensive validation on classification and generation tasks, we demonstrate that SEMU achieves competitive performance compared to state-of-the-art methods in machine unlearning. Moreover, it surpasses current approaches in efficiency, altering fewer model parameters and eliminating the need for the remaining dataset, making it an effective and data-efficient solution.

2. SVD Disentanglement for Efficient Machine Unlearning (SEMU)

To overcome the challenges of gradient-based unlearning, we propose a novel, theoretically grounded method for selecting the most significant subspace of weights, $\theta_\mathbf{s}$, derived from the forgetting dataset $\mathcal{D}_f$.

Our approach employs Singular Value Decomposition (**SVD**) to derive an orthogonal projection onto a lower-dimensional, critical subspace of weights for each layer of a neural network. By normalizing singular values (or eigenvalues), we further refine these subspaces, retaining only the most relevant directions. The level of dimensionality reduction is controlled by a single hyperparameter γ in the range $[0, 1]$. This framework forms the basis of our proposed method, **Singular Value Decomposition for Efficient Machine Unlearning (SEMU)**, presented in Figure 2.

2.1. Singular Value Decomposition

In the context of deep learning models, certain properties of SVD are relevant. Since the layers of neural networks are typically represented as linear operators in $\mathbb{R}^N$, we can simplify Eq. 16 from Theorem C.1 (Horn & Johnson, 2012) to the following form:

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\mathbf{T}, \tag{1}$$

where the parameters $\sigma_1, \ldots, \sigma_R$ are the positive square roots of the eigenvalues of $\mathbf{A}\mathbf{A}^\mathbf{T}$ and $\mathbf{A}^\mathbf{T}\mathbf{A}$, ordered in decreasing magnitude.

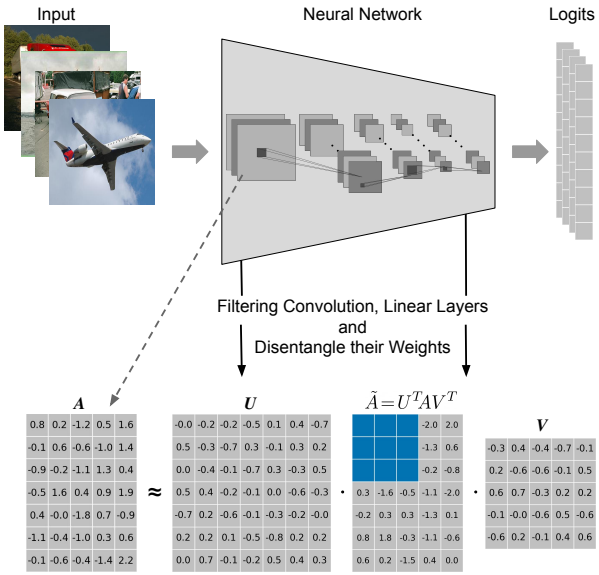

Figure 2. The image illustrates the process of fine-tuning our pretrained model using unlearning data. The model is analyzed with a focus on its convolutional and linear layers. The weight matrices of these layers undergo a process called "weight disentanglement," where only a small subset of parameters within the matrix $\tilde{A}$ is modified. These modified parameters are represented by colored (blue) empty cells in the matrix. A portion of the $\tilde{A}$ matrix remains unchanged, as indicated by the gray cells filled with numbers. During the fine-tuning process, the other matrices, U and V, remain unaltered and they are derived from the gradient projection.

Truncated SVD. Although computing the full SVD of a linear operator $\mathbf{A} \in \mathcal{M}_{n,m}$ is computationally expensive, with a complexity of $\mathcal{O}(nm\min(n, m))$, we can instead focus on a more efficient alternative: the **truncated SVD**. This approach seeks a low-rank approximation $\tilde{\mathbf{A}}$ (rank $r \ll R$) of the original matrix $\mathbf{A}$.

The truncated SVD is expressed as:

$$\mathbf{A} \approx \tilde{\mathbf{A}} = \mathbf{U_r}\mathbf{\Sigma_r}\mathbf{V_r^T}, \tag{2}$$

where $\mathbf{\Sigma_r} \in \mathcal{M}_{r,r}$, $\mathbf{U_r} \in \mathcal{M}_{n,r}$, and $\mathbf{V_r} \in \mathcal{M}_{m,r}$.

This approach allows for a computationally efficient approximation of SVD, while retaining the most significant components of $\mathbf{A}$. By focusing on the largest singular values, the truncated SVD captures the most critical structure in the data, making it especially useful in high-dimensional applications.

2.2. Singular Value Decomposition for Efficient Machine Unlearning

Consider a pretrained model $\theta_\mathbf{o}$ as a sequence of L linear operators (layers) $\theta_\mathbf{o} = \{\mathbf{A_i}\}_{i=1}^L$, where each operator $\mathbf{A_i}$ acts on the input of its corresponding layer x_i as $x_i \to \mathbf{A_i}x_i$. Our goal is to disentangle each of these operators into subspaces such that the gradient of a loss function is concentrated in only a specific part of the such space. In

other words, we aim to find the null space of a linear operator $\mathbf{A_i}$ with respect to its gradient.

Following the gradient-based MU procedure, the gradient of the forgetting loss $-\ell_f(\theta; \mathcal{D}_f)$ with respect to the model weights θ under the forgetting dataset $\mathcal{D}_f$ is given by:

$$\nabla_\theta \left(-\ell_f(\theta; \mathcal{D}_f)|_{\theta=\theta_o}\right). \tag{3}$$

This gradient can be divided into components corresponding to each layer $\mathbf{A_i}$ of the neural network as:

$$G_i = \nabla_{\theta_i} \left(-\ell_f(\theta; \mathcal{D}_f)|_{\theta_i \subseteq \theta_o}\right), \tag{4}$$

where G_i is the gradient for the weights of the i-th layer.

To disentangle a specific operator $\mathbf{A}$, we aim to project it as follows:

$$\mathbf{A} = \mathbf{U}\mathbf{U}^\mathbf{T}\mathbf{A}\mathbf{V}^\mathbf{T}\mathbf{V}, \tag{5}$$

where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices. Intuitively, the goal is to find matrices $\mathbf{U}$ and $\mathbf{V}$ such that, for any corresponding gradient matrix $\mathbf{G}$, the gradient information is concentrated into as few coefficients as possible.

The matrices $\mathbf{U}$ and $\mathbf{V}$ can be efficiently obtained via SVD projection (*see* Eq. 16) of the gradient matrix $\mathbf{G}$:

$$\mathbf{G} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\mathbf{T}, \tag{6}$$

where $\mathbf{\Sigma}$ contains the singular values that indicate the relative importance of each direction.

Selecting most important subspace of $\mathbf{\Sigma}$. While the aforementioned approach effectively provides a projection onto a lower-dimensional subspace, we can refine it further by focusing on the crucial directions. These directions are indicated by the largest eigenvalues of the square matrix $\mathbf{G}\mathbf{G}^\mathbf{T}$ corresponding to the largest singular values σ_i of $\mathbf{G}$, $i = 0 \dots R$. By truncating the $\mathbf{\Sigma}$ matrix, we isolate dominant directions.

Formally, given truncated matrices $\mathbf{A_r} \in \mathbb{R}^{n\times r}$ and $\mathbf{B_r} \in \mathbb{R}^{m\times r}$, we define the subspace $\mathbf{S^r_{A,B}}$ as:

$$\mathbf{S^r_{A,B}} = \{AXB^T : X \in \mathbb{R}^{r\times r}\}. \tag{7}$$

This subspace has a dimensionality of r^2 and allows for efficient computation of the orthogonal projection onto $\mathbf{S^r_{A,B}}$. Specifically, the projection operator is defined as:

$$p_{A,B}(X) = A[A^TXB]B^T \text{ for } X \in \mathbb{R}^{n\times n}, \tag{8}$$

where $p_{A,B}(X)$ represents the orthogonal projection with respect to the Frobenius scalar product on the space of matrices in $\mathbf{S^r_{A,B}}$. This projection is particularly useful when applied to the gradient matrix $\mathbf{G}$. Additionally, performing truncated SVD on $\mathbf{G}$ yields the optimal solution, as guaranteed by Theorem 2.1.

To formalize the optimality of truncated SVD in identifying the most important subspaces, we provide the following theorem:

Theorem 2.1. *Let G denote the gradient matrix. Let U_r, Σ_r and V_r be obtained through the truncated SVD decomposition on G. Then*

$$U_r, V_r = \arg\min_{A,B} d(G; S^r_{A,B}), \tag{9}$$

where d is the distance in terms of the Frobenius metric.

As truncated SVD provides the best low-rank matrix approximation, the next challenge is the selection of the hyperparameter r *a priori*. It determines the rank of the approximation and may vary significantly across layers and also lacks interpretability.

To address this, we note that truncated SVD can be derived by retaining only the largest singular values in Σ. This is equivalent to selecting the top r eigenvalues of GG^T.

Inspired by Principal Component Analysis (PCA), we introduce the concept of *explained variance*, defined as:

$$e_k = \frac{\sum_{j=1}^{k} \sigma_j^2}{\sum_{i=1}^{R} \sigma_i^2}, \tag{10}$$

where σ_i is the *i-th* singular value of G in descending order.

Using this measure, we aim to select the smallest r such that the explained variance exceeds a given threshold γ:

$$r = \arg\min_k e_k \geq \gamma. \tag{11}$$

Due to the fact that e_k is normalized ($e_k \in [0, 1]$), γ becomes an interpretable factor to determine the rank of the approximation.

Unlearning procedure To implement the unlearning procedure, we modify each layer A_i of the neural network θ_o as follows:

$$A_i + U_{i,r}R_iV^T_{i,r}, \tag{12}$$

where $U_{i,r}$ and $V^T_{i,r}$ are obtained via SVD on the corresponding gradient matrix G_i, with r selected based on the hyperparameter γ. R_i **is the only trainable $r \times r$-dimensional matrix**, initialized to $R_i = 0$.

Projection gradient improvement. To minimize the negative impact of unlearning procedure on other weights of the existing model θ_o, we aim to update the weights in a direction perpendicular to the existing weights. For this, we first project the gradient matrix G onto the subspace perpendicular to A using:

$$G_{\perp A} = G - \frac{\langle G, A\rangle}{\|A\|^2}A. \tag{13}$$

We then apply the SVD to the modified gradient matrix $G_{\perp A}$ rather than G.

Information on unlearning objectives (loss functions) as well as exact algorithms are presented in the Appendix.

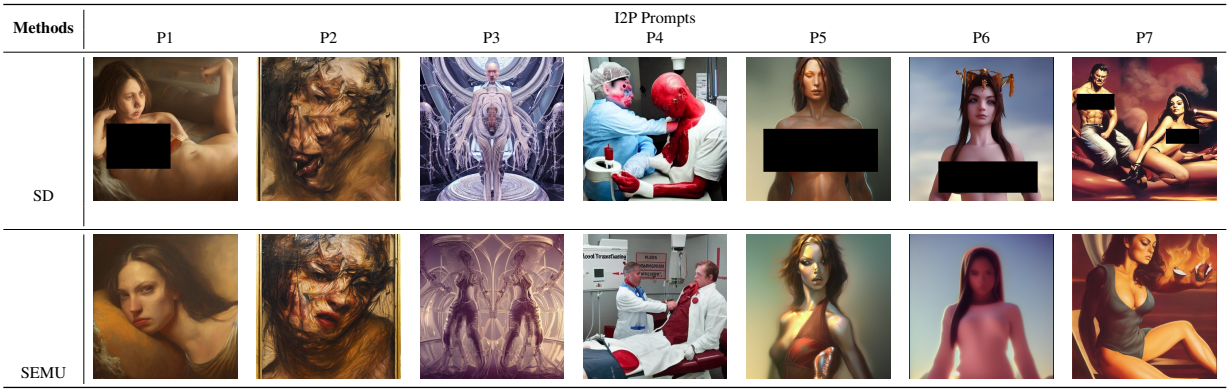

Figure 3. Examples of generated images using Stable Diffusion (top row) and SEMU (bottom row), which generates samples removing *nudity* concept, while preserving the samples semantically close to the original model.

3. Results

We present the detailed description on experimental setup in Appendix G, and the results for image classification problems in Appendix H. We evaluate SEMU on image generation tasks in both class and concept unlearning settings. Our experiments cover two diffusion model architectures: DDPM and Stable Diffusion, applied to CIFAR-10 and Imagenette datasets.

For class unlearning, we first consider the CIFAR-10 dataset using a pretrained conditional DDPM model. The model is fine-tuned to remove its ability to generate a selected class (e.g., airplanes) while maintaining comparable performance for the remaining classes. We compare SEMU against state-of-the-art unlearning methods, including ESD and SalUn, using a fully retrained model as the gold standard. We evaluate SEMU in three different scenarios: (1) without access to the remaining dataset $\mathcal{D}_r$; (2) with full access to $\mathcal{D}_r$; and (3) with access to only a small subset of $\mathcal{D}_r$ (similar to a replay buffer).

Table 6 presents results across key metrics, including Unlearning Accuracy (UA), Task Accuracy (TA), and Fréchet Inception Distance (FID), along with the percentage of trainable parameters used in each method. Our findings indicate that SEMU achieves competitive or superior UA performance while requiring only a small fraction of the trainable parameters (*see* Fig.K.1 and Fig. K.1). We hypothesize that the best trade-off is achieved when SEMU has access to a limited subset of the remaining dataset.

Next, we scale our experiments to Stable Diffusion and perform class-wise forgetting for each class in the Imagenette dataset. We compare SEMU against SalUn, ESD, and FMN, evaluating performance with and without access to $\mathcal{D}_r$ (see Table 7).

Our results show that SEMU performs comparably to state-of-the-art methods across most Imagenette classes, particularly in terms of UA. While SEMU is on par with or slightly behind SalUn and ESD, it significantly outperforms FMN. Notably, SEMU requires only a small number of trainable parameters and does not depend on access to remaining dataset samples in its standard configuration. However, access to $\mathcal{D}_r$ generally improves UA performance, though its impact on FID is less consistent.

Moreover, the images in Figure 3 show that after SEMU Stable Diffusion generates images of similar composition to the original ones but without harmful concepts. We observe that SEMU is semantically closer to the SD's samples than competitive methods (*see* Appendix J).

In the concept unlearning setting, we focus on preventing Stable Diffusion from generating NSFW images, specifically those containing *nudity*. To evaluate this, we generate samples using Stable Diffusion with and without unlearning, using a subset of "dangerous" I2P (Schramowski et al., 2023) prompts from those used by Fan et al. (2023). Our results demonstrate that SEMU effectively removes nudity from generated images. However, when no samples from $\mathcal{D}_r$ are used during unlearning, the sample quality may degrade in certain cases.

Our experiments demonstrate that SEMU is a competitive unlearning method for image generation, achieving results comparable to state-of-the-art approaches while updating only a small subset of parameters. Unlike other methods, SEMU is a general framework that does not rely on model-specific tricks, such as architectural modifications tailored to Stable Diffusion.

4. Conclusions

In this work, we introduce SEMU, a Machine Unlearning method that uses Singular Value Decomposition (SVD) and gradients to identify critical weights that need to be modified. Our experiments demonstrate that SEMU performs competitively with state-of-the-art methods while altering less than 1% of the model's parameters. Consequently SEMU outperforms other methods in preserving the original expression of the model, and we showcase SEMU's ability to erase knowledge from generative models in a real-world use case.

References

Bae, S., Kim, S., Jung, H., and Lim, W. Gradient surgery for one-shot unlearning on generative model. *ICML Workshop on Generative AI & Law*, Jun 2023. URL https://arxiv.org/abs/2307.04550.

Bourtoule, L., Chandrasekaran, V., Choquette-Choo, C. A., Jia, H., Travers, A., Zhang, B., Lie, D., and Papernot, N. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pp. 141–159. IEEE, 2021.

Chen, M., Gao, W., Liu, G., Peng, K., and Wang, C. Boundary Unlearning: Rapid Forgetting of Deep Networks via Shifting the Decision Boundary . In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7766–7775, Los Alamitos, CA, USA, June 2023a. IEEE Computer Society. doi: 10.1109/CVPR52729.2023.00750. URL https://doi.ieeecomputersociety.org/10.1109/CVPR52729.2023.00750.

Chen, M., Gao, W., Liu, G., Peng, K., and Wang, C. Boundary Unlearning: Rapid Forgetting of Deep Networks via Shifting the Decision Boundary . In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7766–7775, Los Alamitos, CA, USA, June 2023b. IEEE Computer Society. doi: 10.1109/CVPR52729.2023.00750. URL https://doi.ieeecomputersociety.org/10.1109/CVPR52729.2023.00750.

Chen, R., Yang, J., Xiong, H., Bai, J., Hu, T., Hao, J., FENG, Y., Zhou, J. T., Wu, J., and Liu, Z. Fast model debias with machine unlearning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023c. URL https://openreview.net/forum?id=BL9Pc7xsdX.

Chourasia, R. and Shah, N. Forget unlearning: towards true data-deletion in machine learning. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

Chundawat, V. S., Tarun, A. K., Mandal, M., and Kankanhalli, M. Zero-shot machine unlearning. *IEEE Transactions on Information Forensics and Security*, 18:2345–2354, 2023. doi: 10.1109/TIFS.2023.3265506.

Fan, C., Liu, J., Zhang, Y., Wong, E., Wei, D., and Liu, S. Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation. *arXiv preprint arXiv:2310.12508*, 2023.

Fan, C., Liu, J., Zhang, Y., Wong, E., Wei, D., and Liu, S. Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=gn0mIhQGNM.

Gandikota, R., Materzynska, J., Fiotto-Kaufman, J., and Bau, D. Erasing concepts from diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2426–2436, 2023.

Golatkar, A., Achille, A., and Soatto, S. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9304–9312, 2020.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Horn, R. A. and Johnson, C. R. *Matrix analysis*. Cambridge university press, 2012.

Howard, J. and Gugger, S. Fastai: a layered api for deep learning. *Information*, 11(2):108, 2020.

Izzo, Z., Smart, M. A., Chaudhuri, K., and Zou, J. Approximate data deletion from machine learning models. In *International Conference on Artificial Intelligence and Statistics*, pp. 2008–2016. PMLR, 2021.

Jia, J., Liu, J., Ram, P., Yao, Y., Liu, G., Liu, Y., Sharma, P., and Liu, S. Model sparsity can simplify machine unlearning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=0jZH883i34.

Kodge, S., Saha, G., and Roy, K. Deep unlearning: Fast and efficient gradient-free class forgetting. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=BmI5p6wBi0.

Kong, Z. and Chaudhuri, K. Data Redaction from Conditional Generative Models . In *2024 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pp. 569–591, Los Alamitos, CA, USA, April 2024. IEEE Computer Society. doi: 10.1109/SaTML59370.2024.00035. URL https://doi.ieeecomputersociety.org/10.1109/SaTML59370.2024.00035.

Kurmanji, M., Triantafillou, P., Hayes, J., and Triantafillou, E. Towards unbounded machine unlearning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=OveBaTtUAT.

Li, G., Hsu, H., Chen, C.-F., and Marculescu, R. Machine unlearning for image-to-image generative models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=9hjVoPWPnh.

Moon, S., Cho, S., and Kim, D. Feature unlearning for pre-trained gans and vaes. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(19): 21420–21428, Mar. 2024. doi: 10.1609/aaai.v38i19.30138. URL https://ojs.aaai.org/index.php/AAAI/article/view/30138.

Neel, S., Roth, A., and Sharifi-Malvajerdi, S. Descent-to-delete: Gradient-based methods for machine unlearning. In Feldman, V., Ligett, K., and Sabato, S. (eds.), *Proceedings of the 32nd International Conference on Algorithmic Learning Theory*, volume 132 of *Proceedings of Machine Learning Research*, pp. 931–962. PMLR, 16–19 Mar 2021. URL https://proceedings.mlr.press/v132/neel21a.html.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Schramowski, P., Brack, M., Deiseroth, B., and Kersting, K. Safe latent diffusion: Mitigating inappropriate degeneration in diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22522–22531, 2023.

Sekhari, A., Acharya, J., Kamath, G., and Suresh, A. T. Remember what you want to forget: Algorithms for machine unlearning. *Advances in Neural Information Processing Systems*, 34:18075–18086, 2021.

Sun, C., Wang, R., Zhang, Y., Jia, J., Liu, J., Liu, G., Liu, S., and Yan, Y. Forget vectors at play: Universal input perturbations driving machine unlearning in image classification, 2025. URL https://arxiv.org/abs/2412.16780.

Sun, H., Zhu, T., Chang, W., and Zhou, W. Generative adversarial networks unlearning, 2023. URL https://arxiv.org/abs/2308.09881.

Tarun, A. K., Chundawat, V. S., Mandal, M., and Kankanhalli, M. Deep regression unlearning. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 33921–33939. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/tarun23a.html.

Thudi, A., Deza, G., Chandrasekaran, V., and Papernot, N. Unrolling SGD: Understanding Factors Influencing Machine Unlearning . In *2022 IEEE 7th European Symposium on Security and Privacy (EuroSP)*, pp. 303–319, Los Alamitos, CA, USA, June 2022. IEEE Computer Society. doi: 10.1109/EuroSP53844.2022.00027. URL https://doi.ieeecomputersociety.org/10.1109/EuroSP53844.2022.00027.

Warnecke, A., Pirch, L., Wressnegger, C., and Rieck, K. Machine unlearning of features and labels. *arXiv preprint arXiv:2108.11577*, 2021.

Zhang, G., Wang, K., Xu, X., Wang, Z., and Shi, H. Forget-me-not: Learning to forget in text-to-image diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1755–1764, 2024.

A. Preliminaries on Machine Unlearning

A.1. Machine Unlearning setup

MU has gained attention due to the rise of foundational models, which, despite their generative capabilities, can unintentionally produce harmful or illegal content. While blocking harmful prompts or retraining models from scratch is theoretically possible (to exclude problematic data, e.g., content lacking copyright permission), these approaches are often impractical due to their high computational cost.

MU provides a more efficient solution by allowing the removal of specific data points, classes, or concepts from a model without full retraining. Its goal is to effectively erase the influence of a *forgetting dataset* while preserving the model's performance. The resulting model should closely match the one that would be retrained on a *remaining dataset* where the forgetting dataset is excluded.

Formally, let us consider a training dataset $\mathcal{D} = \{\mathbf{z}_i\}_{i=1}^N$, consisting of N samples, where $\mathbf{z}_i = (\mathbf{x}_i, \mathbf{y}_i)$ in supervised learning. We define the forgetting dataset as $\mathcal{D}_f \subseteq \mathcal{D}$, with the remaining dataset being its complement, $\mathcal{D}_r = \mathcal{D} \setminus \mathcal{D}_f$.

Let $\theta_\mathbf{o}$ denote the original model trained on $\mathcal{D}$ using a standard training. Following prior research, retraining the model parameters θ from scratch on $\mathcal{D}_r$ serves as the gold standard for MU (Kurmanji et al., 2023). Therefore, the primary objective of MU methods is to derive an **unlearned model** $\theta_\mathbf{u}$ from $\theta_\mathbf{o}$, effectively removing the influence of $\mathcal{D}_f$ while being a computationally efficient alternative to Retrain. Depending on the MU method, the deriving procedure has access to $\mathcal{D}_f$ and/or $\mathcal{D}_r$ dataset.

In this work, we consider the following vision tasks: image classification and image generation.

Image Classification. In this scenario we can typically define two settings: random data forgetting and class-wise forgetting. The setting depends on how the forgetting dataset $\mathcal{D}_f$ is constructed. In the first scenario, the goal is to remove the influence of randomly selected data points from the training set, simulating cases such as the removal of content that lacks copyright permissions. In the second, the objective is to eliminate the influence of an entire class.

Here, we adopt standard MU evaluation metrics to assess the model performance. Specifically, we evaluate

- *unlearning accuracy* (**UA**): 1 - *accuracy* of the unlearned model $\theta_\mathbf{u}$ on the forgetting dataset $\mathcal{D}_f$;
- *membership inference attack* (**MIA**): a privacy metric measuring the vulnerability of $\theta_\mathbf{u}$ to MIA on $\mathcal{D}_f$;
- *remaining accuracy* (**RA**): accuracy of $\theta_\mathbf{u}$ on the remaining dataset $\mathcal{D}_r$;
- *testing accuracy* (**TA**): accuracy of $\theta_\mathbf{u}$ on a test set.

Image Generation with Conditional Diffusion Models. In this work, we focus on two popular classes of conditional diffusion models: denoising diffusion probabilistic models (DDPMs) (Ho et al., 2020) with classifier-free guidance and stable diffusion (Rombach et al., 2022), which are based on the latent diffusion models (LDMs).

To better understand MU in the image generation, we first provide an overview of the diffusion process. Let $\epsilon_\theta(\mathbf{x_t}|c)$ represent the noise estimator parameterized by θ and conditioned on c, where $\mathbf{x_t}$ denotes the data (or latent features) corrupted by noise at step t in the forward diffusion process. The objective of ϵ_θ is to estimate the noise in the reverse diffusion process. Here, the condition c can be an image class (in DDPMs), or a text prompt or concept (in LDMs).

The diffusion process is defined as:

$$\hat{\epsilon}_\theta(\mathbf{x}_t|c) = (1-w)\epsilon_\theta(\mathbf{x}_t|\emptyset) + w\epsilon_\theta(\mathbf{x}_t|c), \tag{14}$$

where $\hat{\epsilon}_\theta(\mathbf{x_t}|c)$ is the noise estimator conditioned on c, $w \in [0,1]$ is the guidance factor, and $\epsilon_\theta(\mathbf{x_t}|\emptyset)$ is the unconditional noise estimator. For inference (image generation), the process begins with Gaussian noise $z_T \sim \mathcal{N}(0,1)$. Using the noise estimator $\hat{\epsilon}_\theta(\mathbf{x_T}|c)$, the model iteratively denoises to obtain $z_{T-1}, z_{T-2}, \ldots$, eventually producing the generated image $z_{t=0}$.

A.2. Challenges in Machine Unlearning

MU faces several significant challenges that should be addressed before it can be effectively adopted by practitioners. These challenges are particularly pressing due to the constantly evolving issues posed by foundational models.

One of them is **the lack of unlearning stability and generality**, which gradient-based methods such as SalUn (Fan et al., 2024) recently attempted to mitigate. These methods identify a subset of parameters, $\theta_{\mathbf{s}} \subseteq \theta_{\mathbf{o}}$, that are most relevant for the forgetting dataset $\mathcal{D}_f$, based on the gradient magnitude of a loss. Fine-tuning this subset with standard MU methods, such as random labeling (RL) and gradient ascent (GA), optimize the unlearning objective using data from $\mathcal{D}_f$, and often $\mathcal{D}_r$.

Although these methods are effective, they introduce challenges that have not been extensively researched. First, **estimating the size of the parameter subset $\theta_{\mathbf{s}}$ is difficult**, relying on heuristics, computationally intensive analyses, or arbitrary choices (e.g., SalUn modifies 50% of parameters). This often leads to unnecessarily large parameter subsets. Second, **large-scale fine-tuning increases computational cost and instability**, especially in image generation, negatively impacting the generalization. These increased costs arise from fine-tuning on the full remaining dataset $\mathcal{D}_r$, which is impractical for large foundation models.

B. Related Works

Numerous unlearning approaches are grounded in knowledge distillation. In (Kurmanji et al., 2023), the authors introduce a teacher-student framework to selectively forget specific data while retaining other instances across various scenarios. Their approach incorporate a rewinding mechanism to obscure the identification of deleted instances. Similarly, in (Chundawat et al., 2023), a distillation-based method is proposed for scenarios where no training data is available to the algorithm. In (Kong & Chaudhuri, 2024), another distillation-driven technique is tailored for conditional generative models, though its evaluation is primarily focused on text-to-image and text-to-speech applications.

A recent study, (Sun et al., 2025), introduces the concept of *forget vectors*, which perturb input data without altering the original model weights. However, this method is specifically designed for image classification and is not applicable to generative models. Meanwhile, the authors of (Kodge et al., 2024) propose a singular value decomposition-based approach that diverges from SEMU in its methodology. By analyzing the activation of samples from the forget and retain classes, they estimate the corresponding feature spaces and quantify the mutual information. Subsequently, they adjust the weights to suppress activations specific to the targeted class. Nevertheless, this method is not designed to handle the unlearning of arbitrary subsets of data.

The authors of (Jia et al., 2023) utilize model sparsification through weight pruning to minimize the discrepancy between an approximate unlearning model and a model retrained from scratch. Similarly, in (Thudi et al., 2022), the authors aim to reduce this discrepancy by introducing a standard deviation loss. In (Fan et al., 2024), the SalUn approach is proposed, which leverages a weight saliency map that can be applied independently or in conjunction with other unlearning methods. SalUn identifies the most influential weights, referred to as salient weights, based on the forgetting loss and prioritizes parameter updates on these weights. It is considered in classification and generation.

Several methods focus on the unlearning of generative models. For instance, (Li et al., 2024) is designed for models that reconstruct images from incomplete inputs, such as masked autoencoders (MAEs), vector-quantized GANs, or diffusion models. Similarly, (Moon et al., 2024) is tailored for both GANs and VAEs, while (Sun et al., 2023) and (Bae et al., 2023) are specifically dedicated to GANs and VAEs.

Another approach, (Chen et al., 2023a), shifts focus from modifying network parameters, to adjusting the decision boundary of the class targeted for forgetting, similar to adversarial attack strategies. Variants of gradient descent methods have also been proposed for unlearning tasks, as exemplified by (Neel et al., 2021). In (Chourasia & Shah, 2023), the authors propose a noisy gradient descent-based solution and argue that indistinguishability from retraining does not guarantee deletion privacy due to residual internal data states. Meanwhile, (Tarun et al., 2023) introduces an unlearning method for deep regression and forecasting models, and (Chen et al., 2023c) addresses the removal of biases from models using counterfactual samples.

C. SVD.

In Theorem C.1, we present the formal definition of Singular Value Decomposition from (Horn & Johnson, 2012).

Theorem C.1. *Let consider a linear operator $\mathbf{A} \in \mathcal{M}_{n,m}$, let $q = min\{m, n\}$, and suppose that R is a rank of $\mathbf{A}$.*

1. *There are unitary matrices $\mathbf{U} \in \mathcal{M}_n$ and $\mathbf{V} \in \mathcal{M}_m$, and a square diagonal matrix*

$$\mathbf{\Sigma}_q = \begin{bmatrix} \sigma_1 & & 0 \\ & \ddots & \\ 0 & & \sigma_q \end{bmatrix} \tag{15}$$

such that $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_R \geq 0 = \sigma_{R+1} = \cdots = \sigma_q$ *and*

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^* \tag{16}$$

in which: $\mathbf{\Sigma} = \mathbf{\Sigma_q}$ *if* $m = n$*;* $\mathbf{\Sigma} = \begin{bmatrix} \mathbf{\Sigma_q} & 0 \end{bmatrix} \in \mathcal{M}_{n,m}$ *if* $m > n$*; and* $\mathbf{\Sigma} = \begin{bmatrix} \mathbf{\Sigma_q} \\ 0 \end{bmatrix} \in \mathcal{M}_{n,m}$ *if* $m < n$*.*

2. *The parameters* $\sigma_1, \ldots, \sigma_R$ *are the positive square roots of the decreasingly ordered nonzero eigenvalues of* $\mathbf{A}\mathbf{A}^*$*, which are the same as the decreasingly nonzero eigenvalues of* $\mathbf{A}^*\mathbf{A}$*.*

D. Proof of Theorem 4.1.

Proof. Observe that every element of $S^r_{A,B}$ is of rank at most r. By the matrix approximation lemma, known as the *Eckart–Young–Mirsky theorem*, the optimal approximation of a matrix G in the Frobenius norm among the rank-r matrices is given by U_r, Σ_r,V_r^T, where the decomposition $\mathbf{G} = \mathbf{U}\mathbf{\Sigma}\mathbf{V^T}$ is the SVD of G. □

E. Unlearning losses.

To finetune the set of parameters $\{R_i\}_{i=1}^L$ for unlearning, we follow the random labeling unlearning losses proposed by SalUn. Specifically, we use the classification loss L_c:

$$\min_{\Delta\theta} \; L_c(\theta_\mathrm{u})\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_\mathrm{f},y'\neq y}\left[\ell_\mathrm{CE}(\theta_\mathrm{u};\mathbf{x},y')\right] + \alpha\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_\mathrm{r}}\left[\ell_\mathrm{CE}(\theta_\mathrm{u};\mathbf{x},y)\right], \tag{17}$$

where ℓ_CE denotes the cross-entropy loss, and α controls the contribution of the remaining dataset $\mathcal{D}_r$.

For the generation task, we apply the generation loss L_g:

$$\min_{\Delta\theta} \; L_g(\theta_\mathrm{u})\mathbb{E}_{(\mathbf{x},c)\sim\mathcal{D}_\mathrm{f},t,\epsilon\sim\mathcal{N}(0,1),c'\neq c}\left[\|\epsilon(\mathbf{x}_t|c') - \epsilon(\mathbf{x}_t|c)\|_2^2\right] + \beta\ell_\mathrm{MSE}(\theta_\mathrm{u};\mathcal{D}_\mathrm{r}), \tag{18}$$

where ℓ_MSE is the mean squared error loss, and β controls the contribution of $\mathcal{D}_r$.

Moreover, our method effectively handles both situations, i.e. with access to the remaining dataset $\mathcal{D}_r$ and without it. In the case where the remaining dataset is unavailable, the situation is equivalent to setting $\alpha = 0$ and $\beta = 0$.

F. Algorithms for SEMU unlearning.

In this, we present the pseudo-codes for each algorithm used for SEMU. The Section is structured as follows – firstly, we present the general procedure for selecting weights in SEMU, Alg. 1. Then, we introduce using SEMU in a classification setting (2), and in the image generation one (3).

Algorithm 1 Pseudo code of SEMU selecting weights procedure.

Require: Forgetting set $\mathcal{D}_f$, original model θ_o, explanation parameter γ, and forgetting loss function ℓ.
procedure SEMU_WEIGHTS_SELECTION($\mathcal{D}_f$, θ_o, γ, ℓ)
  $\mathbf{g} \leftarrow \varnothing$ ▷ Array of accumulated gradients corresponding to θ_o
  for $\mathbf{b} \leftarrow$ all batches of $\mathcal{D}_f$ **do**
    $\mathbf{g} \leftarrow \mathbf{g} + \nabla_{\theta_o}\left(-\ell(\theta_o; \mathbf{b})\right)$ ▷ Accumulating gradients according to Eq. 4
  end for
  $\theta_{\mathbf{c}} \leftarrow \varnothing$ ▷ Original model with changed layers
  $\theta_{\mathbf{u}} \leftarrow \varnothing$ ▷ Trainable parameters for unlearning procedure
  for $layer \leftarrow 1 \ldots L$ **do**
    $g_l \leftarrow g|_{\theta_{\mathbf{l}} \subseteq \theta_{\mathbf{o}}}$ ▷ Selecting gradients corresponding to l-*th* layer
    $g_{l\perp\theta_{\mathbf{l}}} \leftarrow g_l - \frac{\langle g_l, \theta_{\mathbf{l}} \rangle}{\|\theta_{\mathbf{l}}\|^2}\theta_{\mathbf{l}}$ ▷ Perpendicular projection onto the l-*th* layer weights space $\theta_{\mathbf{l}}$ from Eq. 13
    $U_l, \Sigma_l\, V_l \leftarrow \mathrm{SVD}(\,(g_{l\perp\theta_{\mathbf{l}}}))$ ▷ **SVD** projection on the $g_{l\perp\theta_{\mathbf{l}}}$ via Eq. 6
    $r_l \leftarrow \arg\min_k e_k \geq \gamma$ ▷ Selecting the low-rank r_l in the SVD projection with Eq. 11
    $R_{l,r_l} \leftarrow \mathbf{0}$ ▷ Initializing the trainable parameters
    $\theta_{\mathbf{l}} \leftarrow \theta_{\mathbf{l}} + U_{l,r_l} R_{l,r_l} V_{l,r_l}^T$ ▷ Updating the l-*th* layer parameters with truncated SVD matrices (Eq. 12)
    $\theta_{\mathbf{c}} \leftarrow \theta_{\mathbf{c}} \cup \theta_{\mathbf{l}}$ ▷ Updating original model
    $\theta_{\mathbf{u}} \leftarrow \theta_{\mathbf{u}} \cup R_{l,r_l}$ ▷ Updating the set of trainable parameters
  end for
  return $\theta_{\mathbf{c}}, \theta_{\mathbf{u}}$

Algorithm 2 Pseudo code of SEMU in classification tasks.

Hyper-parameters: learning rate η, explanation parameter γ, forgetting loss function ℓ, and number of epochs E.
Require: Relabeled forgetting set $\mathcal{D}'_f = \{(\mathbf{x}_i, c')|(\mathbf{x}_i, c_i) \in \mathcal{D}_f, c' \neq c_i\}$
$\theta_o, \theta_u \leftarrow$ SEMU_WEIGHTS_SELECTION(D_f, θ_o, γ, ℓ) ▷ Updating θ_o and setting trainable parameters θ_u with Alg. 1
$\mathcal{D}' \leftarrow \mathcal{D}'_f \cup \varnothing \left(\mathcal{D}' \leftarrow \mathcal{D}'_f \cup \mathcal{D}_r\right)$ ▷ When using *retrain* mode
for $epoch \leftarrow 0 \ldots E-1$ **do**
  for $\mathbf{b} \leftarrow$ all batches of $\mathcal{D}'$ **do**
    $\mathbf{g} \leftarrow \nabla_\theta L_c(\theta; \mathbf{b})|_{\theta=\theta_u}$ ▷ Batch-wise loss from eq. 17
    $\theta_u \leftarrow \theta_u - \eta\mathbf{g}$ ▷ One step SGD
  end for
end for
return

Algorithm 3 Pseudo code of SEMU in generation tasks.

Hyper-parameters: learning rate η, explanation parameter γ, forgetting loss function ℓ, and number of iterations T.
Require: Relabeled forgetting set $\mathcal{D}'_f = \{(\mathbf{x}_i, c')|(\mathbf{x}_i, c_i) \in \mathcal{D}_f, c' \neq c_i\}$
  $\theta_o, \theta_u \leftarrow$ SEMU_WEIGHTS_SELECTION$(\mathrm{D}_f, \theta_o, \gamma, \ell)$ ▷ Updating θ_o and setting trainable parameters θ_u with Alg. 1
  $\mathcal{D}' \leftarrow \mathcal{D}'_f \cup \varnothing \left(\mathcal{D}' \leftarrow \mathcal{D}'_f \cup \mathcal{D}_r\right)$ ▷ When using *retrain* mode
  for $it \leftarrow 0 \ldots T-1$ **do**
    Sampling batch $\mathbf{b}$ from $\mathcal{D}'$
    $\mathbf{g} \leftarrow \nabla_\theta L_g(\theta; \mathbf{b})|_{\theta=\theta_u}$ ▷ Batch-wise loss from eq. 18
    $\theta_u \leftarrow \theta_u - \eta \mathbf{g}$ ▷ One step SGD
  end for
  return

G. Experimental setup details.

G.1. Image Classification.

Following the methodology outlined by Fan et al. (2023), we conduct a series of experiments to evaluate the performance of data forgetting in image classification tasks. Specifically, we focus on two scenarios of random data forgetting: 10% and 50% of the training data. These experiments are performed on widely used datasets, namely CIFAR-10 and CIFAR-100, and employ popular deep learning architectures such as ResNet-18 and VGG-16. Additionally, we explore a class-wise forgetting setup in image classification, where we specifically target the removal of entire classes from the training data. For this task, we utilize the ResNet-18 architecture and the CIFAR-10 dataset. Within experiments we run grid search to find the best parameter $\gamma \in [60\% - 95\%]$ and report the best performing model. We compare SEMU with FT (Warnecke et al., 2021), RL (Golatkar et al., 2020), GA (Thudi et al., 2022), IU (Izzo et al., 2021), $\ell 1$-sparse (Jia et al., 2023), 2 boundary unlearning methods (Chen et al., 2023b), boundary shrink (BS) and boundary expanding (BE).

G.2. Image generation.

Similarly to the image classification setup, we conduct extensive experiments following the evaluation procedure presented in Fan et al. (2023). Specifically, we consider diffusion models, which are the current state-of-the-art methods for image generation. Our experiments encompass two distinct families of diffusion models: denoising diffusion probabilistic models (DDPMs) and latent diffusion models (LDMs), such as Stable Diffusion. DDPMs operate directly in image space, which makes them suitable for lower-dimensional generation tasks but limits their applicability to high-resolution images. In contrast, LDMs employ a pretrained autoencoder to encode images into a lower-dimensional latent space, enabling scalable high-resolution generation. We evaluate our method in two scenarios:

Class Unlearning In this setting, we aim to remove a specific class from a pretrained diffusion model. For DDPM, we attempt to unlearn the "airplane" class from CIFAR-10. For Stable Diffusion, we unlearn each class from the Imagenette dataset (Howard & Gugger, 2020), a subset of ImageNet containing ten high-resolution categories. We measure the effectiveness of unlearning using Unlearning Accuracy (UA) on generated samples from the forgotten class and evaluate the impact on generation quality by computing the Fréchet Inception Distance (FID) on the remaining classes. Our approach is compared against ESD (Gandikota et al., 2023) and SalUn for DDPM, and against ESD, SalUn, and FMN (Zhang et al., 2024) for Stable Diffusion.

Concept Unlearning. Here, we focus on forgetting a broad concept rather than a specific class. We choose the NSFW concept of nudity as the target for unlearning in Stable Diffusion. To evaluate effectiveness, we first generate 800 images of both nude and clothed individuals using Stable Diffusion. After applying our unlearning method, we assess the model's ability to generate images conditioned on a subset of I2P's "dangerous" prompts. Additionally, we measure its effectiveness in reducing NSFW generation across the full I2P dataset of harmful prompts (Schramowski et al., 2023).

For all experiments, we report results using the best-performing hyperparameters. We determine the explained variance threshold γ based on empirical trade-offs:

- **DDPM:** $\gamma \in [0.9, 0.95]$ for all layers;
- **Stable Diffusion:** $\gamma = 1.0$ for cross-attention layers, and $\gamma \in [0.9, 0.95]$ for all other layers.

We found that cross-attention layers are particularly sensitive to lower values of γ, as reducing their rank too aggressively leads to a loss of critical information—specifically, the ability to associate one concept with another. Importantly, setting $\gamma = 1.0$ still results in a low-rank decomposition, as it retains all directions corresponding to nonzero singular values, preserving the original rank of the matrix.

H. Results for SEMU in image classification task.

Image classification. We present results for the random data-forgetting, considering 10% and 50% of the data, in the following tables: Table 1 for CIFAR100 with ResNet18, Table 2 for CIFAR10 with ResNet18, and Table 3 for CIFAR10 with VGG-16. Additionally, Table 4 shows results for class-wise forgetting on CIFAR10 with ResNet18.

The results demonstrate that SEMU can successfully perform unlearning by altering even less than 1% of the model weights (only for CIFAR100 it is a bit more than 1%). Furthermore, SEMU achieves the smallest gap in Testing Accuracy (TA), indicating that the model remains largely unchanged from its initial state. This minimal impact on test set accuracy suggests that SEMU preserves the model's ability to perform with high fidelity, even after unlearning. What is more, even class-wise setup requires alteration of less than 1% weights to achieve unlearning (see Table 4).

We also evaluate SEMU when it has access to the remaining dataset ($SEMU_{remain}$) to compare its performance with other machine unlearning methods under the same experimental setup. When the remaining data is available, SEMU achieves slightly better results. This indicates that, with a properly designed method, access to the remaining dataset is not strictly necessary, offering computational savings while maintaining strong performance.

Lastly, we examine how SalUn, as the most similar approach, performs under conditions of reduced data availability and decreased saliency sparsity (10% compared to the default 50%) in Table 5 and Figure 4. Notably, even a slight reduction in the remaining dataset negatively impacts SalUn's performance. Additionally, decreasing the proportion of altered parameters to just 10% further decreases the model's effectiveness. Those results show that SEMU is much more robust than competing approaches.

Table 1. Comparison of methods for Random Data Forgetting (10% and 50%) on ResNet-18 with CIFAR-100 dataset. The table reports Unlearning Accuracy (UA), Remaining Accuracy (RA), Testing Accuracy (TA), and Membership Inference Attack (MIA), with values in parentheses showing differences from the *Retrain* baseline. TParams denotes the percentage of trained parameters relative to standard ResNet-18 (not unforgetting). SEMU is the only method that achieves as close target accuracy as the retrain method while altering the smallest portion of the model, meaning that the MU with SEMU is not changing the model a lot. Note that we bold results achieving the closest TA accuracy to Retrain and those which alter the smallest portion of model's weigths.

Methods	Random Data Forgetting (10%)					Random Data Forgetting (50%)				
	UA	RA	TA	MIA	TParams	UA	RA	TA	MIA	TParams
Retrain	26.47	99.97	74.13	51.00	100%	32.69	99.99	67.22	61.15	100%
FT	2.42 (24.05)	99.95 (0.02)	75.55 (1.42)	11.04 (39.96)	100%	2.71 (29.98)	99.96 (0.03)	75.11 (7.89)	10.71 (50.44)	100%
RL	55.03 (28.56)	99.81 (0.16)	70.03 (4.09)	98.97 (47.97)	100%	50.52 (17.83)	99.47 (0.52)	56.75 (10.47)	95.91 (34.76)	100%
GA	3.13 (23.34)	97.33 (2.64)	75.31 (1.18)	7.24 (43.76)	100%	2.61 (30.08)	97.49 (2.50)	75.27 (8.05)	5.92 (55.23)	100%
IU	3.18 (23.29)	97.15 (2.82)	73.49 (0.64)	9.62 (41.38)	100%	12.64 (20.05)	87.96 (12.03)	62.76 (4.46)	17.54 (43.61)	100%
BE	2.31 (24.16)	97.27 (2.70)	73.93 (0.20)	9.62 (41.38)	100%	2.76 (29.93)	97.39 (2.60)	74.05 (6.83)	8.85 (52.30)	100%
BS	2.27 (24.20)	97.41 (2.56)	75.26 (1.13)	5.82 (45.18)	100%	2.99 (29.70)	97.24 (2.75)	73.38 (6.16)	8.76 (52.39)	100%
ℓ_1-sparse	10.64 (15.83)	96.62 (3.35)	70.99 (3.14)	22.58 (28.42)	100%	39.86 (7.17)	78.17 (21.82)	55.65 (11.57)	40.43 (20.72)	100%
SalUn	27.53 (1.06)	97.00 (2.97)	67.79 (6.34)	70.79 (19.79)	50%	26.17 (6.52)	94.04 (5.95)	61.39 (5.83)	59.47 (1.68)	50%
SalUn-soft	24.24 (2.23)	98.95 (1.02)	70.48 (3.65)	79.13 (28.13)	50%	23.26 (9.43)	98.32 (1.67)	63.08 (4.14)	77.90 (16.75)	50%
SEMU	2.53 (23.94)	97.39 (2.58)	**74.14 (0.01)**	8.82 (42.18)	**1.18%**	3.80 (28.89)	96.44 (3.55)	71.24 (4.02)	12.25 (48.90)	**1.18%**
$SEMU_{remain}$	2.93 (23.54)	97.33 (2.64)	74.16 (0.03)	11.93 (39.07)	**1.18%**	7.92 (24.77)	92.37 (7.62)	**67.16 (0.06)**	17.11 (44.04)	1.44%

Table 2. Comparison of methods for Random Data Forgetting (10% and 50%) on ResNet-18 with CIFAR-10. The table reports Unlearning Accuracy (UA), Remaining Accuracy (RA), Testing Accuracy (TA), and Membership Inference Attack (MIA), with values in parentheses showing differences from the *Retrain* baseline. TParams denotes the percentage of trained parameters relative to standard ResNet-18 (not unforgetting). Note that we bold results achieving the closest TA accuracy to Retrain and those which alter the smallest portion of model's weigths.

Methods	Random Data Forgetting (10%)					Random Data Forgetting (50%)				
	UA	RA	TA	MIA	TParams	UA	RA	TA	MIA	TParams
Retrain	5.24 (0.00)	100.00 (0.00)	94.26 (0.00)	12.88 (0.00)	100%	7.91 (0.00)	100.00 (0.00)	91.72 (0.00)	19.29 (0.00)	100%
FT	0.63 (4.61)	99.88 (0.12)	94.06 (0.20)	2.70 (10.19)	100%	0.44 (7.47)	99.96 (0.04)	94.23 (2.52)	2.15 (17.14)	100%
RL	7.61 (2.37)	99.67 (0.33)	92.83 (1.43)	37.36 (24.47)	100%	4.80 (3.11)	99.55 (0.45)	91.31 (0.40)	41.95 (22.66)	100%
GA	0.69 (4.56)	99.50 (0.50)	94.01 (0.25)	1.70 (11.18)	100%	0.40 (7.50)	99.61 (0.39)	94.34 (2.63)	1.22 (18.07)	100%
IU	1.07 (4.17)	99.20 (0.80)	93.20 (1.06)	2.67 (10.21)	100%	3.97 (3.94)	96.21 (3.79)	90.00 (1.71)	7.29 (12.00)	100%
BE	0.59 (4.65)	99.42 (0.58)	93.85 (0.42)	7.47 (5.41)	100%	3.08 (4.82)	96.84 (3.16)	90.41 (1.31)	24.87 (5.58)	100%
BS	1.78 (3.47)	98.29 (1.71)	92.69 (1.57)	8.96 (3.93)	100%	9.76 (1.85)	90.19 (9.81)	83.71 (8.01)	32.15 (12.86)	100%
ℓ_1-sparse	4.19 (1.06)	97.74 (2.26)	91.59 (2.67)	9.84 (3.04)	100%	1.44 (6.47)	99.52 (0.48)	93.13 (1.41)	4.76 (14.52)	100%
SalUn	2.85 (2.39)	99.62 (0.38)	93.93 (0.33)	14.39 (1.51)	100%	7.75 (0.16)	94.28 (5.72)	89.29 (2.43)	16.99 (2.30)	100%
SalUn-soft	4.19 (1.06)	99.74 (0.26)	93.44 (0.83)	19.49 (6.61)	100%	3.41 (4.49)	99.62 (0.38)	91.82 (0.11)	31.50 (12.21)	100%
SEMU	0.60 (4.64)	99.40 (0.60)	**94.22 (0.04)**	5.40 (7.48)	**0.54%**	1.77 (6.14)	98.12 (1.88)	91.80 (0.08)	7.20 (12.09)	**0.64%**
SEMU$_{remain}$	0.69 (4.55)	99.43 (0.57)	**94.30 (0.04)**	5.51 (7.37)	**0.54%**	1.82 (6.09)	98.12 (1.88)	**91.72 (0.00)**	7.54 (11.75)	0.72%

Table 3. Comparison of methods for Random Data Forgetting (10% and 50%) on VGG-16 with CIFAR-10 dataset. The table reports Unlearning Accuracy (UA), Remaining Accuracy (RA), Testing Accuracy (TA), and Membership Inference Attack (MIA), with values in parentheses showing differences from the *Retrain* baseline. TParams denotes the percentage of trained parameters relative to standard VGG-16 (not unforgetting). Note that we bold results achieving the closest TA accuracy to Retrain and those which alter the smallest portion of model's weigths.

Methods	Random Data Forgetting (10%)					Random Data Forgetting (50%)				
	UA	RA	TA	MIA	TParams	UA	RA	TA	MIA	TParams
Retrain	5.98	99.99	93.06	10.36	100%	9.47	100.00	90.18	16.64	100%
FT	1.51 (4.47)	99.54 (0.45)	92.64 (0.42)	3.76 (6.60)	100%	5.70 (3.77)	97.51 (2.49)	89.37 (0.81)	12.20 (4.44)	100%
RL	5.71 (0.27)	99.65 (0.34)	92.29 (0.77)	15.98 (5.62)	100%	4.09 (5.38)	96.77 (3.23)	89.91 (0.27)	13.88 (2.76)	100%
GA	0.93 (5.05)	99.37 (0.62)	93.63 (0.57)	1.36 (9.00)	100%	0.63 (8.84)	99.38 (0.62)	93.64 (3.46)	1.15 (15.49)	100%
IU	1.69 (4.29)	98.78 (1.21)	91.69 (1.37)	2.71 (7.65)	100%	5.71 (3.76)	94.56 (5.44)	87.23 (2.95)	8.34 (8.30)	100%
BE	0.80 (5.18)	99.39 (0.60)	93.68 (0.62)	1.42 (8.94)	100%	20.58 (11.11)	79.40 (20.60)	72.58 (17.60)	11.74 (4.90)	100%
BS	0.80 (5.18)	99.40 (0.59)	93.68 (0.62)	1.38 (8.98)	100%	2.44 (7.03)	97.56 (2.44)	89.69 (0.49)	4.90 (11.74)	100%
ℓ_1-sparse	4.98 (1.00)	97.03 (2.96)	90.15 (2.91)	9.69 (0.67)	100%	3.13 (6.34)	98.77 (1.23)	91.01 (0.83)	7.06 (9.58)	100%
SalUn	3.89 (2.09)	98.74 (1.25)	91.62 (1.44)	9.96 (0.40)	100%	3.02 (6.45)	98.14 (1.86)	89.82 (0.36)	15.15 (1.49)	100%
SalUn-soft	5.24 (0.74)	99.70 (0.29)	92.26 (0.80)	12.31 (1.95)	100%	3.44 (6.03)	99.64 (0.36)	91.11 (0.93)	16.19 (0.45)	100%
SEMU	0.67 (5.31)	99.33 (0.66)	**93.09 (0.03)**	5.02 (5.34)	0.89%	3.31 (6.16)	96.32 (3.68)	90.01 (0.17)	18.92 (2.28)	0.34%
SEMU$_{remain}$	0.62 (5.36)	99.37 (0.62)	93.27 (0.21)	7.02 (3.34)	**0.23%**	2.56 (6.91)	96.98 (3.02)	**90.21 (0.03)**	16.06 (0.58)	**0.29%**

Table 4. Performance evaluation for class-wise forgetting on ResNet-18, pre-trained on the CIFAR-10 dataset. The table presents the results of various methods in terms of Unlearning Accuracy (UA), Remaining Accuracy (RA), Testing Accuracy (TA), and Membership Inference Attack (MIA). The values in parentheses indicate the difference compared to the *Retrain* baseline.

Methods	UA	RA	TA	MIA	TParams
Retrain	100.0	100.0	92.47	100.0	100%
FT	31.69 (68.31)	99.92 (0.07)	94.78 (2.31)	93.53 (6.47)	100%
RL	89.33 (10.67)	99.92 (0.08)	94.52 (2.06)	100.0 (0.00)	100%
GA	99.91 (0.09)	38.92 (61.07)	38.18 (54.29)	99.98 (0.02)	100%
IU	97.02 (2.98)	94.78 (5.22)	89.10 (3.37)	99.13 (0.87)	100%
BE	79.13 (20.87)	97.71 (2.29)	91.88 (0.59)	93.60 (6.40)	100%
BS	79.60 (20.40)	97.79 (2.21)	91.94 (0.52)	93.42 (6.58)	100%
ℓ_1-sparse	100.0 (0.00)	97.92 (2.08)	92.29 (0.18)	100.0 (0.00)	100%
SalUn	99.91 (0.09)	99.93 (0.07)	94.56 (2.09)	100.0 (0.00)	50%
SalUn-soft	97.13 (2.87)	99.88 (0.12)	94.64 (2.18)	100.0 (0.00)	50%
SEMU	99.83 (0.17)	98.22 (1.78)	**92.26 (0.21)**	100.00 (0.00)	0.87%
SEMU$_{remain}$	99.99 (0.01)	99.48 (0.52)	94.76 (2.29)	100.00 (0.00)	**0.63%**

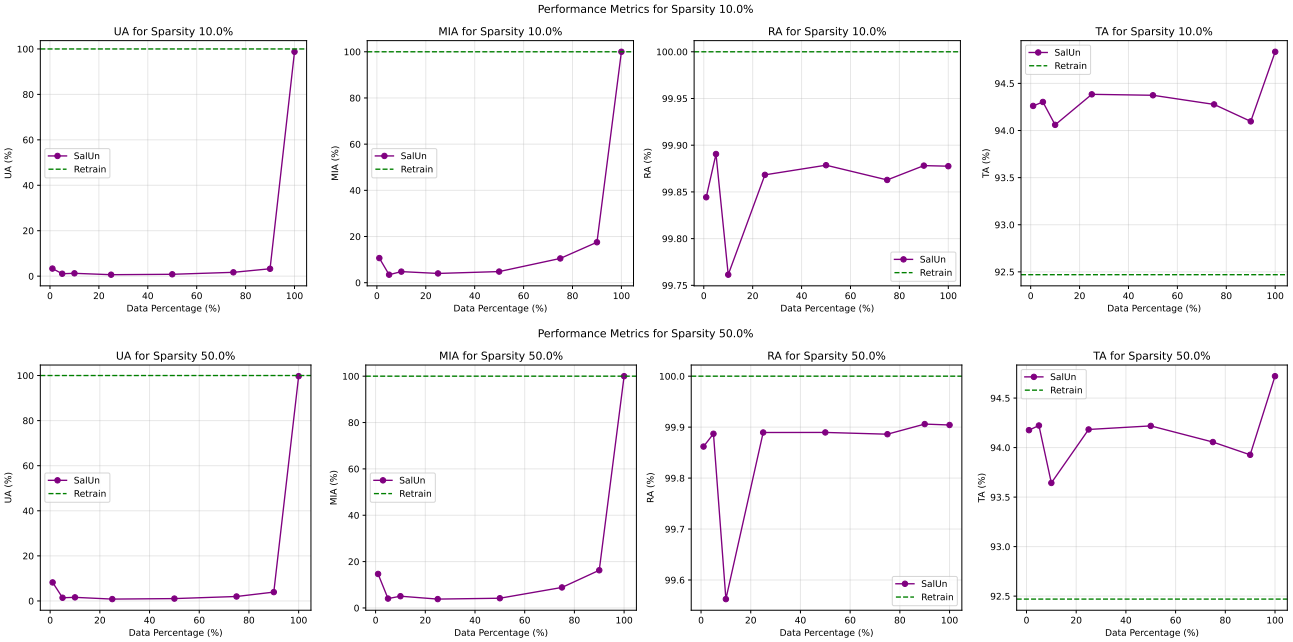

Figure 4. Overview of SalUn results for the Class-Wise Forgetting scenario on ResNet-18 with CIFAR-10 for different percentages of available data from the class selected to forget. The top row depicts results for 10%, while the bottom row shows scores for 50% of the saliency sparsity. The plots in consecutive columns demonstrate Unlearning Accuracy (UA), Membership Inference Attack (MIA), Remaining Accuracy (RA) and Testing Accuracy (TA), respectively. In all cases, results are compared to the *Retrain* baseline.

Table 5. Comparison of SalUn results for Class-Wise Forgetting on ResNet-18 with CIFAR-10 for different percentages of available data from the class selected to forget (from 1 to 100%), and different saliency sparsity (10 and 50%). The table reports Unlearning Accuracy (UA), Remaining Accuracy (RA), Testing Accuracy (TA), and Membership Inference Attack (MIA), with values in parentheses showing differences from the *Retrain* baseline.

Available data	Saliency sparsity (10%)				Saliency Sparsity (50%)			
	UA	RA	TA	MIA	UA	RA	TA	MIA
Retrain	100.00	100.00	92.47	100.00	100.00	100.00	92.47	100.00
1%	3.33 (96.67)	99.84 (0.16)	94.26 (1.79)	10.67 (89.33)	8.22 (91.78)	99.86 (0.14)	94.18 (1.71)	14.67 (85.33)
5%	1.07 (98.93)	99.89 (0.11)	94.30 (1.83)	3.47 (96.53)	1.42 (98.58)	99.89 (0.11)	94.22 (1.75)	4.04 (95.96)
10%	1.24 (98.76)	99.76 (0.24)	94.06 (1.59)	4.80 (95.20)	1.60 (98.40)	99.56 (0.44)	93.64 (1.17)	5.07 (94.93)
25%	0.64 (99.36)	99.87 (0.13)	94.38 (1.91)	4.01 (95.99)	0.83 (99.17)	99.89 (0.11)	94.18 (1.71)	3.82 (96.18)
50%	0.82 (99.18)	99.88 (0.12)	94.37 (1.90)	4.82 (95.18)	1.05 (98.95)	99.89 (0.11)	94.22 (1.75)	4.20 (95.80)
75%	1.68 (98.32)	99.86 (0.14)	94.28 (1.81)	10.51 (89.49)	1.98 (98.02)	99.89 (0.11)	94.06 (1.59)	8.85 (91.15)
90%	3.24 (96.76)	99.88 (0.12)	94.10 (1.63)	17.52 (82.48)	3.93 (96.07)	99.91 (0.09)	93.93 (1.46)	16.26 (83.74)
100%	98.76 (1.24)	99.88 (0.12)	94.84 (2.37)	100.00 (0.00)	99.71 (0.29)	99.90 (0.10)	94.72 (2.25)	100.00 (0.00)

I. Additional results for SEMU in image generation task.

I.1. CIFAR10 generation with DDPM.

Table 6. Class-wise forgetting on classifier-free guidance DDPM.

Methods	**UA** (↑)	**TA** (↑)	**FID** (↓)	**Params** (↓)
Retrain	100.00	100.00	11.69	100%
ESD	**100.00**	**–**	17.37	**–**
SalUn	99.20	14.22	**11.21**	50%
SEMU	95.60	**14.87**	16.93	**1.2%**
SEMU_{subset}	99.40	14.71	13.93	1.5%
SEMU_{remain}	**100.00**	14.64	14.51	1.8%

I.2. ImageNette generation with Stable Diffusion.

Table 7. Performance of class-wise forgetting on Imagenette using SD. The best unlearning performance for each forgetting class is highlighted in **bold** for UA and FID, respectively. For SEMU, we averaged FIDs for the smaller number of classes due to biasing of FID metric.

Forget. Class	SEMU_{remain}		SEMU		SalUn		ESD		FMN	
	UA (↑)	FID (↓)	UA (↑)	FID (↓)	UA (↑)	FID (↓)	UA (↑)	FID (↓)	UA (↑)	FID (↓)
Tench	89.00	11.40	94.00	2.31	**100.00**	2.53	99.40	**1.22**	42.40	1.63
English Springer	94.00	4.14	94.00	2.27	**100.00**	**0.79**	100.00	1.02	27.20	1.75
Cassette Player	98.00	1.36	92.00	26.23	99.80	0.91	**100.00**	1.84	93.80	**0.80**
Chain Saw	96.00	8.54	64.00	1.12	**100.00**	1.58	96.80	1.48	48.40	**0.94**
Church	85.00	14.30	70.00	–	**99.60**	**0.90**	98.60	1.91	23.80	1.32
French Horn	**100.00**	**0.81**	98.00	4.20	**100.00**	**0.94**	99.80	1.08	45.00	0.99
Garbage Truck	99.00	2.51	86.00	–	**100.00**	**0.91**	100.00	2.71	41.40	0.92
Gas Pump	98.00	2.48	88.00	1.32	**100.00**	**1.05**	100.00	1.99	53.60	1.30
Golf Ball	95.00	5.77	84.00	1.46	98.80	1.45	**99.60**	**0.80**	15.40	1.05
Parachute	95.00	13.85	68.00	–	**100.00**	1.16	99.80	**0.91**	34.40	2.33
Average	94.90	6.52	83.80	5.56 *	**99.82**	**1.22**	99.40	1.49	42.54	1.30

J. Samples of nudity unlearning with different machine unlearning methods.

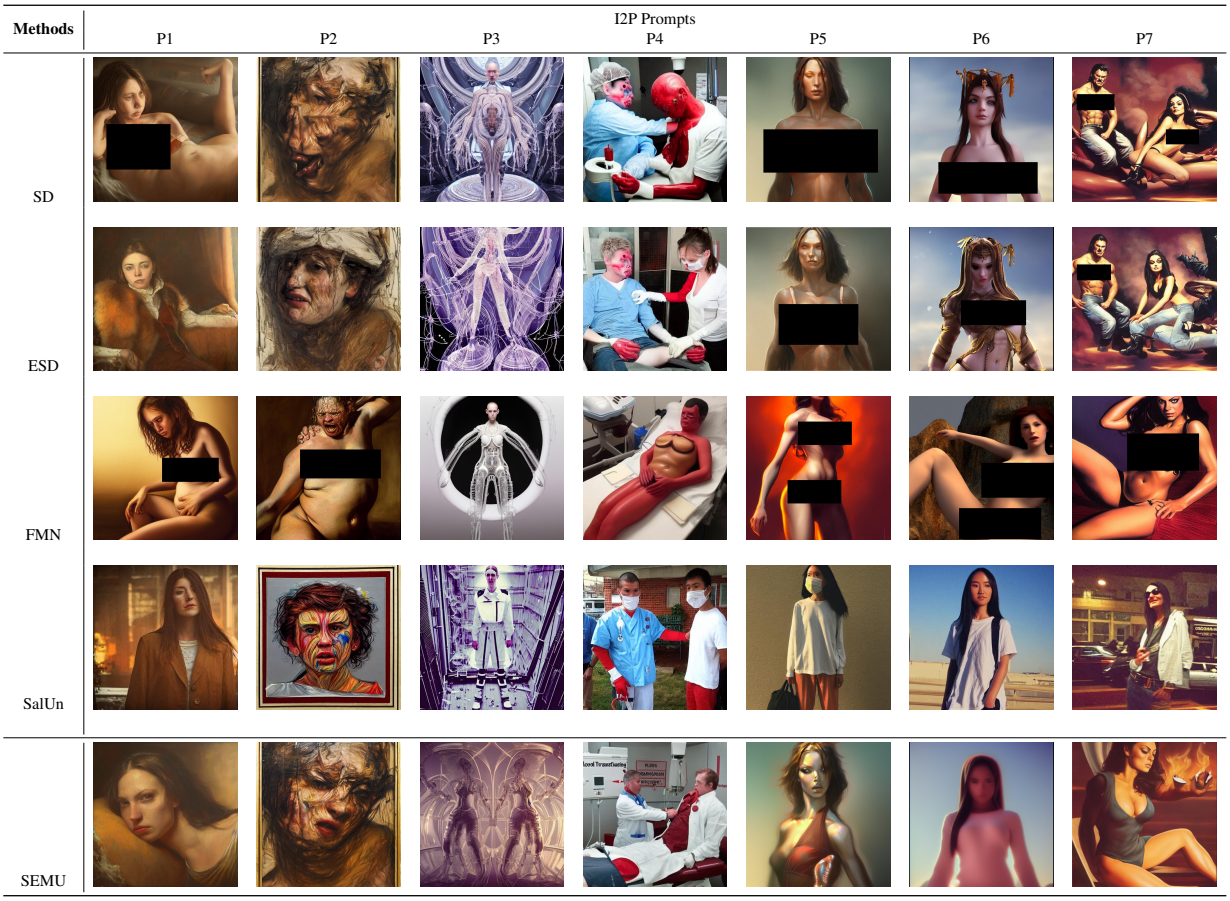

Figure 5. Examples of generated images using Stable Diffusion and different machine unlearning methods. The samples for ESD, FMN, and SalUn are from Fan et al. (2023). SEMU is presented in the bottom row and generates samples removing *nudity* concept, while preserving the samples semantically closer to the original model, SD (top row), than the competitve solution SalUn.

K. Samples from the diffusion models unlearned with SEMU.

In this Section, we present the samples from the DDPM model unlearned a CIFAR10's class (*airplanes*) with SEMU.

K.1. Samples from DDPM on CIFAR10.

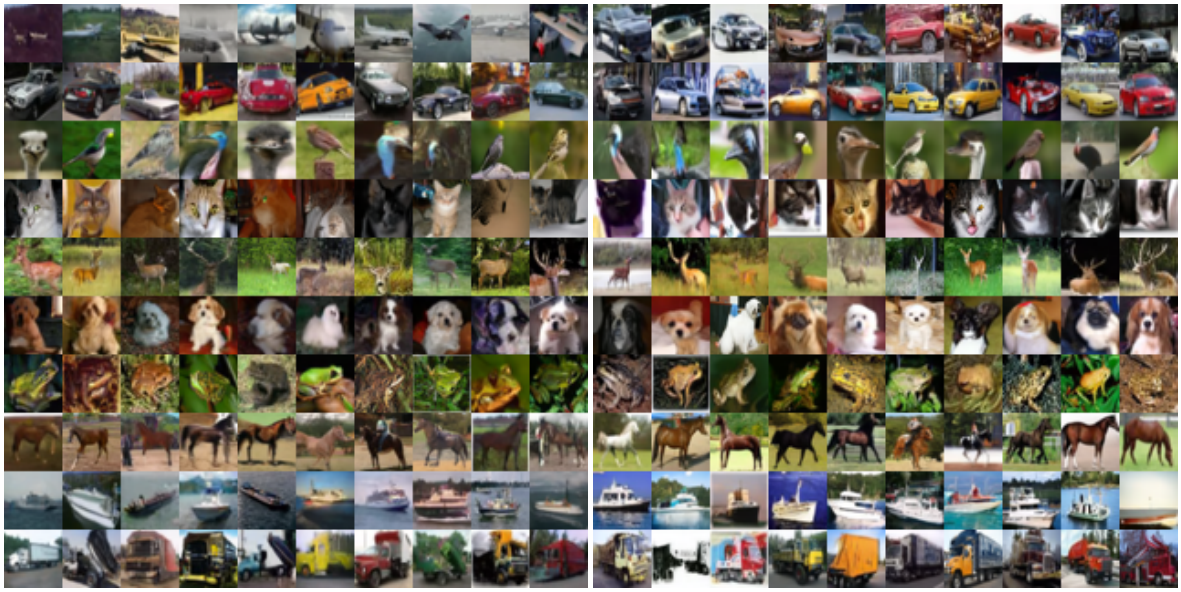

Figure 6. Comparison between SEMU with (**left**) and without (**right**) access to the remaining dataset. The DDPM model was pretrained on CIFAR10, and with SEMU, we unlearned the class *airplanes* (top rows). We observe that the access to the remaining dataset stabilizes generation and helps to change samples from one class to the other. On the other hand, lack of such an access prevent model from total forgetting.

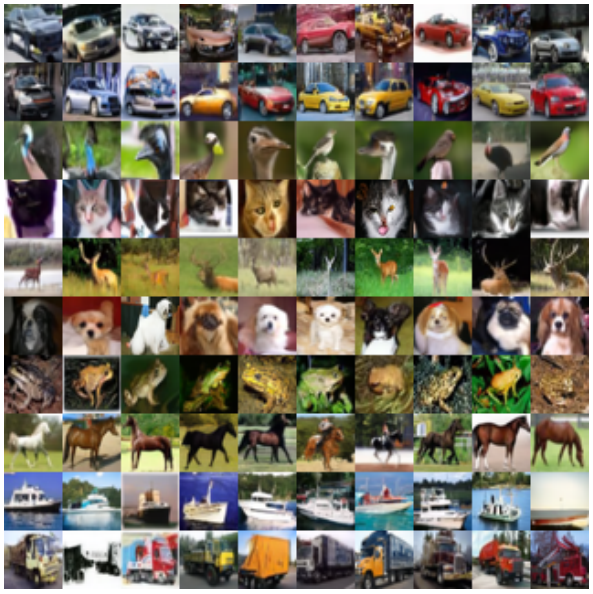

Figure 7. Setting in which the unlearned model have an access to the very limited number of samples from the remaining dataset. As we can see, a limited number of additional datapoints is a sufficient for SEMU to have the same quality of samples as the when having access to the whole remaining dataset.