Editing Memory of Deep Neural Networks with Emphasis on Eigen Arithmetic

Rohit Gandikota
Northeastern University
gandikota.ro@northeastern.edu

Abstract

Editing memory of large neural network models has been gaining much attention with the recent advancements in Large Language Models (LLMs) and diffusion models. From copyright image take-downs to inappropriate image restrictions, memory editing has various crucial applications for safer AI practices. In this work, we address the task of memory editing in vision models through interventions. We also show that the attention heads in diffusion models act as a memory bank for concepts. By editing the knowledge of the models through intervention, one can avoid the overhead of expensive retraining of the large models. Our code, data, and results can be found at https://github.com/rohitgandikota/eigens

1 Introduction

Understanding deep neural networks has been active research in the recent decade. There are works that proposed visualizing the filters through deconvolution Zeiler and Fergus (2014). The initial findings of interpretability have shown progressive learning of filters indicating a hierarchy of filterting Forsyth et al. (1999).

Recent advances in Large Language Models (LLMs) Radford et al. (2018); Ouyang et al. (2022); Touvron et al. (2023); Hoffmann et al. (2022); Scao et al. (2022) and diffusion models Rombach et al. (2022a); Ramesh et al. (2021); Saharia et al. (2022) has enabled for text, image, and video generations that are hyper-realistic. With such advancements, there is a need for safe, robust, and fair generative models. There are recent works Schramowski et al. (2022); Bedapudi (2022); Laborde (2022); Rando et al. (2022) that restrict such unsafe data post-generation. But with open-sourcing of the models, it is getting trickier to restrict such content. This calls for a change in model parameters such that it is harder to produce such harmful/unsafe generations.

Recent works have explored the method of filtering training samples such that the model is not trained on such content Nichol et al. (2021). However, this is an expensive task, and almost impossible to filter out all the content from a training population (5 billion images in case of stable diffusion Schuhmann et al. (2022)). In addition to that, there is also the overhead of training. We propose editing the pre-trained models by

simply editing the activations of neurons. This would be a simple in-expensive fine-tuning compared to the previous solutions. This would also ensure the knowledge is edited within the parameters, restricting the unsafe generations at level-0.

As the cost of training grows, there has been increasing interest in lightweight model-editing methods that alter the behavior of large-scale generative models given little or no new training data. In text generators, a model's knowledge of facts can be edited based on a single statement of the fact by modifying specific neurons Dai et al. (2022) or layers Meng et al. (2022), or by using hypernetworks De Cao et al. (2021); Mitchell et al. (2021). In image synthesis, a generative adversarial network (GAN) can be edited using a handful of words Gal et al. (2022), a few sketches Wang et al. (2021), warping gestures Wang et al. (2022), or copy-and-paste Bau et al. (2020). Recently, it has been shown that text-conditional diffusion models can be edited by associating a token for a new subject trained using only a handful of images **?**Kumari et al. (2023); Ruiz et al. (2022). Unlike previous methods that add or modify the appearance of objects, the goal of our current work is to erase a targeted visual concept from a diffusion model giving only a single textual description of the concept, object, or style to be removed.

With the increase in training costs, there has been a growing interest in lightweight model-editing techniques that can modify the behavior of large-scale generative models without requiring new training data. In the case of text generators, specific neurons Dai et al. (2022) or layers Meng et al. (2022) can be modified based on a single statement of a fact to alter the model's knowledge of that fact. Alternatively, hyper networks can be used for editing purposes De Cao et al. (2021); Mitchell et al. (2021). Similarly, for image synthesis, a generative adversarial network (GAN) can be edited using a few words Gal et al. (2022), sketches Wang et al. (2021), gestures Wang et al. (2022), or copy-and-paste Bau et al. (2020). Unlike previous methods that modify or add objects, the current research aims to erase a targeted visual concept from a diffusion model using a single textual description of the object, concept, or style to be removed. A recent study has shown model editing to erase concepts from text-to-image diffusion models through the concept of negative guidance Gandikota et al. (2023). We, however, choose the approach of intervention rather than model parameter editing to interpret the location of knowledge in deep neural networks.

2 Method

The goal of the method is to alter the output of the network by intervening in the hidden activations in a certain way so as to specifically target a particular concept while keeping the rest of the concepts unaltered. To elaborate, we collect the activations of the network $\mathcal{N}$ (with total N layers) at layer L. We define the part of the network from input to layer L as $\mathcal{N}^{1:L}$ and the rest of the network as $\mathcal{N}^{L:N}$. Therefore, we collect the intermediate activations at layer L by passing the inputs X through subnetwork $\mathcal{N}^{1:L}$ and collect the outputs.

$$\eta_L = \mathcal{N}^{1:L}(X) \tag{1}$$

We then edit the outputs η_L using a function f, (more detail on the function in further sections) and pass it through the remainder of the network to check the effect on the

model.

$$y_{edit} = \mathcal{N}^{L:N}(f(\eta_L)) \tag{2}$$

The core of this work is to design the function f such that y_{edit} is altered only at certain concepts and the same elsewhere compared to the unedited output y

$$y = \mathcal{N}^{L:N}(\eta_L) \tag{3}$$

2.1 MLP

A fully connected network's last layer can be simply viewed as a logistic regression layer. In other words, the initial layers of MLP act as a feature extractor, and the final layer acts as a decision layer to classify. We intervene in the outputs of the penultimate layer and edit it such that the network's accuracy for a single class has deteriorated while the other classes' accuracies are intact.

Specifically, we intervene in the $\mathcal{N}^{1:N-1}$ layer output and design a function f_{MLP} such that the per-class accuracy of one class is drastically reduced, while the remaining class accuracies are unaltered.

We design the edit function as removing the important direction u_i of a class i from the activations. We calculate the important direction by collecting multiple activations for the inputs corresponding to class i and calculate the eigenvectors using SVD. Let $\eta^1_{N-1}, \eta^2_{N-1}, ..., \eta^m_{N-1}$ be the layer $N-1$ activations of m samples belonging to class i. We calculate the eigenvectors using SVD (singular value decomposition) by:

$$[\eta^1_{N-1}, \eta^2_{N-1}, ..., \eta^m_{N-1}] = U\Sigma V^T \tag{4}$$

We then take the top important direction u_i by selecting the first eigenvector in U.

$$u_i = U[:1] \tag{5}$$

After calculating the important directions of class i, during the inference, we delete this direction from all the activations at layer $N-1$ irrespective of the class.

$$\eta^{edit,i}_{N-1} = \eta_{N-1} - <u_i, \eta_{N-1}> \frac{u_i}{\|u_i\|} \tag{6}$$

Where $<a, b>$ is the dot product of vectors a and b. The above function represents the erasure of the important direction u_i from all the activations at layer $N-1$. We then pass the edited activations through the rest of the network to collect the predictions

$$y_{edit,i} = \mathcal{N}^{N-1:N}(\eta^{edit,i}_{N-1}) \tag{7}$$

This edited output belongs to the edited model activations where important directions of class i have been erased.

2.2 CNN

For CNN, we try repeating the experiment of removing the eigendirections for both convolution outputs and FC outputs. We find that convolution outputs are much more intertwined than the FC layer outputs.

For convolution outputs, we calculate the eigendirections using SVD and select the top direction to delete from the activations. We find that convolution maps are not knowledge banks, instead, they are merely feature extractors in CNN. We find this phenomenon in both shallow and deep networks.

2.3 Diffusion Models

Diffusion models are a class of generative models that learn the distribution space as a gradual denoising process Sohl-Dickstein et al. (2015); Ho et al. (2020). Starting from sampled Gaussian noise, the model gradually denoises for T time steps until a final image is formed. In practice, the diffusion model predicts noise ϵ_t at each time step t that is used to generate the intermediate denoised image x_t; where x_T corresponds to the initial noise and x_0 corresponds to the final image. This denoising process is modeled as a Markov transition probability.

$$p_\theta(x_{T:0}) = p(x_T) \prod_{t=T}^{1} p_\theta(x_{t-1}|x_t) \tag{8}$$

Latent diffusion models (LDM) Rombach et al. (2022b) improve efficiency by operating in a lower dimensional latent space z of a pre-trained variational autoencoder with encoder $\mathcal{E}$ and decoder $\mathcal{D}$. During training, for an image x, noise is added to its encoded latent, $z = \mathcal{E}(x)$ leading to z_t where the noise level increases with t. LDM process can be interpreted as a sequence of denoising models with identical parameters θ that learn to predict the noise $\epsilon_\theta(z_t, c, t)$ added to z_t conditioned on the timestep t as well as a text condition c. The following objective function is optimized:

$$\mathcal{L} = E_{z_t \in \mathcal{E}(x), t, c, \epsilon \sim \mathcal{N}(0,1)}[\|\epsilon - \epsilon_\theta(z_t, c, t)\|_2^2] \tag{9}$$

Classifier-free guidance is a technique employed to regulate image generation, as described in Ho et al. Ho and Salimans (2022). This method involves redirecting the probability distribution towards data that is highly probable according to an implicit classifier $p(c|z_t)$. This approach is used during inference and requires that the model be jointly trained on both conditional and unconditional denoising. The conditional and unconditional scores are both obtained from the model during inference. The final score $\tilde{\epsilon}_\theta(z_t, c, t)$ is then directed towards the conditioned score and away from the unconditioned score by utilizing a guidance scale $\alpha > 1$.

$$\tilde{\epsilon}_\theta(z_t, c, t) = \epsilon_\theta(z_t, t) + \alpha(\epsilon_\theta(z_t, c, t) - \epsilon_\theta(z_t, t)) \tag{10}$$

The inference process starts from a Gaussian noise $z_T \sim \mathcal{N}(0, 1)$ and is denoised with the $\tilde{\epsilon}_\theta(z_T, c, T)$ to get z_{T-1}. This process is done sequentially till z_0 and is transformed to image space using the decoder $x_0 \leftarrow \mathcal{D}(z_0)$.

In this work, we would first explore the eigendirections of the outputs of activation layers. Especially the cross attentions which act like a gateway between the text prompt and image space. We also find that with generative models simply editing the activation lead to a small change in the learned distribution, which in turn leads to degraded image space. Small changes in learned distribution lead to big changes in visual images. Therefore we also show results on finetuning the cross-attention weights using the technique of replacing the semantics of the prompt as shown in Equation 11.

$$\epsilon_\theta(x_t, c, t) \leftarrow \epsilon_\theta(x_t, t) \tag{11}$$

This ideally finetunes the model to guide the concepts towards unconditional images. In other words, we are unlearning the concepts by confusing the network. It is important to note that we do not require any dataset to teach the network as we use the knowledge of the model itself to unlearn a concept.

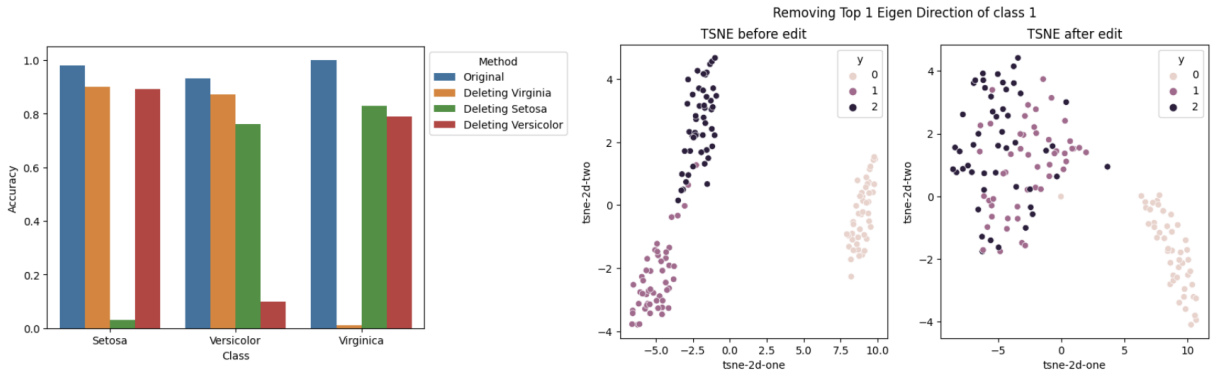

Figure 1: Our method demonstrates that erasing eigen directions of one class disrupts the model's knowledge of the class while keeping the other knowledge intact. The experiment shows that there is interference with other classes which indicates partial entanglement or sharing of directions between classes within the model. On the right, we also visualize how erasing the top eigen direction of one class can affect the activation distribution using TSNE

3 Experiments

3.1 Eigen Directions in FC Layers

We first explore the importance of eigen directions in fully connected layers. For this experiment, we choose the IRIS dataset with IRIS dataset Fisher (1936). The dataset consists of 4 feature samples with 50 training data samples in each of the 3 classes. To test the importance of eigen directions, we choose simple FC-connected layers with 2 hidden layers of size 100 each. After training the network in a regular fashion, we probe the final layer activations and collect samples for each class individually. We then compute the important directions for each class and store them in separate vectors u_1, u_2, u_3. For the test samples, we remove each direction and note the class accuracies, and plot in Figure 1. We can see that removing the eigen directions of a class reduces the accuracy of the corresponding class while keeping the other accuracies intact. This shows the importance of eigen directions of the activation outputs. In other words, for easily separable data, the MLP network tries to disentangle the data such that the final layer can do a simple decision boundary. Disrupting these directions lead to the erasure of the knowledge of the particular class, but not others. We also effect of erasing top-n eigen directions instead of top-1 in the Appendix.

3.2 Eigen Directions in CNN

We demonstrate the erasure of eigendirections in convolution layers as well as FC layers from a CNN. We observe that convolutions are much more entangled between classes showing that CNN is a generic operation where no knowledge is stored, but rather a feature extractor as shown in Table A.4. On the other hand, FC layers perform some partial entanglement but show that they are the knowledge banks in the CNN model architecture. For this experiment, we use CIFAR 10 dataset Krizhevsky et al. (2009). The dataset consists of 10 classes with 32×32 colored images of count

60000. After training the network, we use the Resnet-50 architecture to probe the intermediate convolution and FC activation for each individual class. We then erase these directions from the activations by intervening in the network during inference and noting the classification outputs. We show more results in Appendix including top-n eigen direction erasures.

	Class wise Accuracy									
	plane	**car**	**bird**	**cat**	**deer**	**dog**	**frog**	**horse**	**ship**	**truck**
plane	31.5	7.9	-2.6	-2.1	-2.8	-4.2	2.9	-3.9	32.6	-0.2
car	22.6	10.6	-5.3	-1.2	-1.8	3.4	5.2	0.5	6.3	-7.6
bird	42.6	15.0	-2.6	-2.5	-2.5	-6.8	0.8	-4.2	41.0	8.5
cat	9.5	0.0	3.0	-1.9	-1.5	12.9	0.9	-2.2	-1.3	4.2
deer	51.3	21.9	-4.8	-7.7	-0.9	-9.2	-2.4	3.5	36.8	11.4
dog	-1.8	-2.0	-2.1	-2.8	0.7	12.1	0.0	1.7	5.9	-2.5
frog	8.1	13.3	-9.4	8.7	-10.3	6.8	-6.7	11.1	-4.3	9.8
horse	1.7	-1.2	0.7	-0.8	-2.4	-0.9	-1.3	1.5	1.6	-0.6
ship	40.1	21.3	-2.8	-3.6	-3.9	0.4	2.9	-4.8	36.6	2.3
truck	29.3	11.7	-4.3	-1.2	-3.1	4.7	6.1	-1.0	8.7	-8.1

Table 1: We see that erasing eigen directions from convolution maps has no interpretable effect on the percentage change of the class accuracies compared to the unedited model. There is no pattern in the accuracies indicating heavy entanglement of direction between classes within the model.

	Class wise Accuracy									
	plane	**car**	**bird**	**cat**	**deer**	**dog**	**frog**	**horse**	**ship**	**truck**
plane	-2.1	2.5	-0.9	-3.6	7.6	4.0	-4.7	-3.1	-3.2	18.9
car	-2.8	**91.4**	-0.6	10.6	9.2	-3.9	-4.2	-1.5	-1.5	-1.7
bird	-2.1	-0.1	2.3	-4.3	1.7	6.1	-3.7	-2.4	-1.8	13.9
cat	-2.0	91.0	-2.5	**16.0**	1.8	-0.3	-3.9	-2.7	-2.1	1.7
deer	-2.5	2.2	-2.5	-3.7	**46.7**	0.5	-3.8	-2.2	-3.5	6.0
dog	2.0	-2.7	-2.2	-7.4	-0.7	**77.1**	-2.4	-3.2	-2.7	21.5
frog	-0.2	0.1	-1.4	-3.1	4.8	4.3	-3.0	-3.2	-2.7	25.1
horse	-1.1	-0.3	-1.7	-1.7	-4.2	-0.7	1.9	**89.4**	10.7	-1.7
ship	-2.1	-0.6	-0.3	-0.1	-2.1	-2.2	-0.2	13.6	**85.2**	-1.6
truck	-3.0	-0.9	-0.1	-5.6	7.1	5.1	-3.5	-3.3	-3.4	**86.6**

Table 2: Erasing eigen directions from FC layers of the CNN has an interpretable effect on the class accuracies. For most of the classes, erasing the eigendirection has a significant effect on the corresponding class accuracy (a diagonal effect) as shown in bold.

Figure 2: Erasing eigen directions of the "bear" concept from attention heads leads to concept erasure from the model output. However, the image quality also seems to go down due to this erasure. We show the edited images in the top column and the corresponding edited images after intervention in the bottom column.

3.3 Eigen Directions in Diffusion Models

Recent works have shown that editing cross attentions can be the key to modifying concepts in the generated images Hertz et al. (2022); Kumari et al. (2023). We, therefore, turn our attention to the attention heads of the diffusion model. More specifically, we edit the outputs of the cross-attentions of the model. To understand this, we first visualize individual attention heads that activate a certain concept in the prompt. For example, we look at the attention heads that activate for the token "bear" in the prompt "a furry bear catching fish". In Figure B.1 and B.2, we find that certain attention heads consistently attend to certain concepts. For example, layer 3 seems to attend to the concept of "bear" while layer 12 attends to "car". We believe erasing eigen directions of the concepts from their important attention heads may lead to memory erasure of the concepts from diffusion models.

When erasing the eigen directions from the outputs of the attention heads (attention heads are similar to FC layers, making the method similar as discussed in section 2.1), we find that it does erase the concept from the generated images, but it still has some effect on the images. This is due to diffusion models being image generative models, a slight disturbance in the distribution can lead to a major effect on the output quality of the images as shown in Figure 2

Since there is a need for more finer and careful editing, we take inspiration from Gandikota et al. (2023) by finetuning the cross-attention head weights using guidance using Equation 11. To be more precise, we replace the concept we wish to erase with an unconditional score, therefore guiding the model away from the concept we wish to erase as shown in Figure 3. It is also important to address the specificity of the model

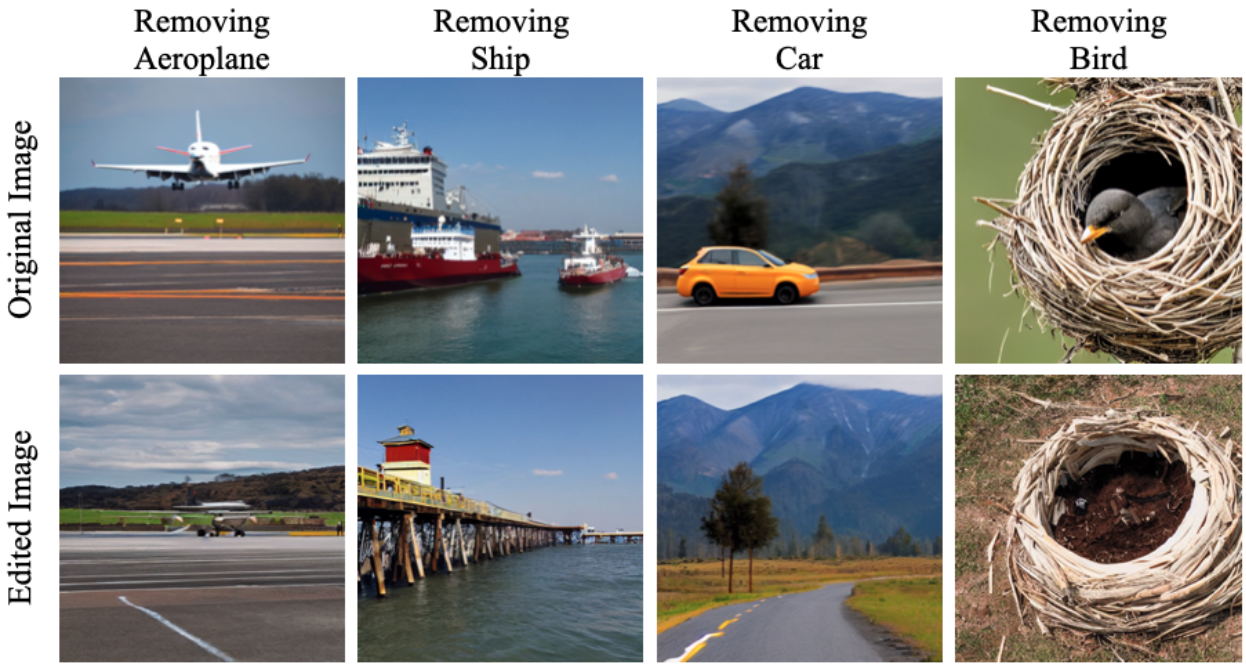

Figure 3: Editing the model weights of diffusion attention heads shows a much finer knowledge editing compared to eigen direction erasure. We find that the concepts are removed and the background is preserved while maintaining higher quality.

after these edits. To elaborate, we need to ensure that by erasing certain concepts from the model, the capability of the model to generate other concepts is not disturbed. To analyze this, we erase a concept from the diffusion model and generate a set of images from various classes. We then classify the images using standard classifier networks that were trained in the previous sections. We find that editing one concept does not lead to the erasure of other concepts from the model. We show a detailed analysis of removal using classifiers and more visual results in the Appendix.

4 Conclusion

This work proposes that editing eigen directions of the intermediate activations in deep neural networks can potentially erase the selective memory of the networks. We show that by removing the eigen directions from the diffusion models can lead to control editing of the model's knowledge which can hugely save re-training costs of the model. We also show that a fast and efficient finetuning that only requires the name of the concept to erase is much more efficient and practical.

We show the efficacy of our method on three different network architectures. Firstly, we show that editing the FC layers in MLP can lead to selective erasure. Secondly, we show that earlier convolution layers act as feature extractors and not memory banks therefore editing these layers does not lead to selective erasures. Lastly, we illustrate our method's capability on attention heads of diffusion models by selectively erasing concepts through eigen direction editing.

References

Bau, D., Liu, S., Wang, T., Zhu, J.-Y., and Torralba, A. (2020). Rewriting a deep generative model. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Bedapudi, P. (2022). NudeNet: Neural nets for nudity detection and censoring.

Dai, D., Dong, L., Hao, Y., Sui, Z., Chang, B., and Wei, F. (2022). Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502.

De Cao, N., Aziz, W., and Titov, I. (2021). Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6491–6506.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188.

Forsyth, D. A., Mundy, J. L., di Gesú, V., Cipolla, R., LeCun, Y., Haffner, P., Bottou, L., and Bengio, Y. (1999). Object recognition with gradient-based learning. *Shape, contour and grouping in computer vision*, pages 319–345.

Gal, R., Patashnik, O., Maron, H., Bermano, A. H., Chechik, G., and Cohen-Or, D. (2022). Stylegan-nada: Clip-guided domain adaptation of image generators. *ACM Transactions on Graphics (TOG)*, 41(4):1–13.

Gandikota, R., Materzyńska, J., Fiotto-Kaufman, J., and Bau, D. (2023). Erasing concepts from diffusion models. *arXiv preprint arXiv:2303.07345*.

Hertz, A., Mokady, R., Tenenbaum, J., Aberman, K., Pritch, Y., and Cohen-Or, D. (2022). Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*.

Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851.

Ho, J. and Salimans, T. (2022). Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D. d. L., Hendricks, L. A., Welbl, J., Clark, A., et al. (2022). Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

Kumari, N., Zhang, B., Zhang, R., Shechtman, E., and Zhu, J.-Y. (2023). Multi-concept customization of text-to-image diffusion.

Laborde, G. (2022). NSFW detection machine learning model.

Meng, K., Bau, D., Andonian, A. J., and Belinkov, Y. (2022). Locating and editing factual associations in gpt. In *Advances in Neural Information Processing Systems*.

Mitchell, E., Lin, C., Bosselut, A., Finn, C., and Manning, C. D. (2021). Fast model editing at scale. In *International Conference on Learning Representations*.

Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., and Chen, M. (2021). Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. (2022). Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. (2018). Improving language understanding by generative pre-training.

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. (2021). Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR.

Rando, J., Paleka, D., Lindner, D., Heim, L., and Tramèr, F. (2022). Red-teaming the stable diffusion safety filter. *arXiv preprint arXiv:2210.04610*.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. (2022a). High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. (2022b). High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., and Aberman, K. (2022). Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. *arXiv preprint arXiv:2208.12242*.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S. K. S., Ayan, B. K., Mahdavi, S. S., Lopes, R. G., et al. (2022). Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*.

Scao, T. L., Fan, A., Akiki, C., Pavlick, E., Ilić, S., Hesslow, D., Castagné, R., Luccioni, A. S., Yvon, F., Gallé, M., et al. (2022). Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

Schramowski, P., Brack, M., Deiseroth, B., and Kersting, K. (2022). Safe latent diffusion: Mitigating inappropriate degeneration in diffusion models. *arXiv preprint arXiv:2211.05105*.

Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., et al. (2022). Laion-5b: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. (2023). Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Wang, S.-Y., Bau, D., and Zhu, J.-Y. (2021). Sketch your own gan. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14050–14060.

Wang, S.-Y., Bau, D., and Zhu, J.-Y. (2022). Rewriting geometric rules of a gan. *ACM Transactions on Graphics (TOG)*, 41(4):1–16.

Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pages 818–833. Springer.

Appendix

In this appendix, we first show the effects of erasing top3 and top5 eigen directions from both FC layers and convolution layers. We then show

A Top-n Eigen Directions

A.1 Convolution Layers

We find that erasing any number of eigen directions from the convolution layers does not show any interpretable effect on the class-wise accuracy scores. We do observe that erasing more eigen directions lead to a worse unlearning effect on the model as a whole which is obvious intuitively. By heavily editing the model activations, we are technically providing random junk data to the model.

	Class wise Accuracy									
	plane	**car**	**bird**	**cat**	**deer**	**dog**	**frog**	**horse**	**ship**	**truck**
plane	6.8	8.1	24.6	25.8	29.0	33.9	31.4	24.0	-0.9	-2.3
car	16.7	7.4	18.1	14.6	5.3	23.5	32.1	15.5	2.9	-6.6
bird	24.7	17.8	10.1	-3.3	3.4	15.3	25.1	1.3	13.3	-7.5
cat	41.5	14.3	14.5	-11.0	5.5	8.7	12.3	-2.6	35.5	1.4
deer	30.3	10.9	3.2	2.2	1.6	14.9	17.9	1.4	28.5	-7.0
dog	17.0	0.2	12.1	-6.7	2.0	16.0	8.2	1.0	22.4	-4.6
frog	36.3	35.1	10.2	2.4	6.6	6.4	10.0	-1.1	15.1	-7.4
horse	12.7	5.2	7.8	3.7	-0.5	6.5	21.0	7.4	8.8	-4.8
ship	-1.6	3.6	23.2	25.2	48.3	31.7	32.0	30.6	7.1	-4.2
truck	15.3	8.5	16.0	17.7	24.6	23.4	31.4	12.9	-1.1	-2.2

Table A.1: Erasing top-3 eigen directions from convolution layers.

	Class wise Accuracy									
	plane	**car**	**bird**	**cat**	**deer**	**dog**	**frog**	**horse**	**ship**	**truck**
plane	11.0	4.4	20.3	21.0	29.5	28.3	32.1	27.2	-3.2	-1.3
car	11.5	6.4	11.4	16.2	18.6	21.4	31.8	17.8	-1.0	-1.5
bird	25.2	10.5	4.8	-3.7	2.4	14.5	23.7	1.9	9.0	-5.7
cat	17.6	-0.5	-0.9	-2.1	-1.3	12.9	19.6	2.8	16.4	-6.7
deer	30.8	7.6	4.2	5.2	3.1	18.1	21.1	0.1	7.5	-1.9
dog	34.1	3.7	11.5	-9.7	5.6	1.3	10.0	-2.3	28.1	4.2
frog	51.9	44.2	4.2	-6.8	-2.8	0.7	-3.1	-2.6	48.1	16.7
horse	21.6	-2.4	2.9	2.5	0.6	8.7	16.2	1.5	12.3	-2.7
ship	0.2	-0.9	25.2	31.6	50.6	36.1	32.5	45.1	-5.5	12.2
truck	10.2	0.7	7.6	21.0	11.6	24.9	30.9	5.2	5.1	-4.2

Table A.2: Erasing top-5 eigen directions from convolution layers.

A.2 FC Layers

By editing the top-3 eigen directions from the FC layers we find some interesting patterns. Specifically, we observe that "car", "dog", "horse" and "truck" are constantly affected by editing the top-3 eigen directions of any class. We believe that these classes have a common most important direction as the top-3 eigen direction of all classes. We find a similar effect on top-5 eigen direction erasures.

Class wise Accuracy										
	plane	**car**	**bird**	**cat**	**deer**	**dog**	**frog**	**horse**	**ship**	**truck**
plane	69.1	91.2	0.7	-16.7	59.7	79.9	-5.9	88.3	68.0	87.5
car	-8.4	91.6	-9.5	72.2	-6.7	79.9	80.3	88.6	71.9	87.3
bird	73.3	91.3	78.6	-8.7	17.5	80.3	-7.9	89.5	-4.5	87.7
cat	-3.6	91.5	-10.8	72.5	-6.8	80.2	79.4	89.5	45.7	71.5
deer	-0.9	91.5	0.6	73.1	85.0	80.2	-11.0	89.4	-4.8	87.7
dog	-8.7	91.5	-5.6	73.5	81.6	80.0	-8.3	87.7	85.2	26.3
frog	0.1	91.3	-11.2	61.9	-6.1	79.9	79.8	89.3	84.4	87.9
horse	0.4	91.4	-1.6	72.7	83.9	80.0	-10.3	89.5	-4.8	87.7
ship	-4.5	91.2	-11.8	72.7	-5.4	79.5	79.4	89.0	84.1	87.2
truck	15.0	91.1	-6.1	-6.9	-6.4	79.7	79.5	89.3	84.8	87.9

Table A.3: Erasing top-3 eigen directions from linear FC layers. We observe that "car", "dog", "horse" and "truck" are constantly affected by editing the top-3 eigen directions of any class. We believe that these classes have a common most important direction as the top-3 eigen direction of all classes.

Class wise Accuracy										
	plane	**car**	**bird**	**cat**	**deer**	**dog**	**frog**	**horse**	**ship**	**truck**
plane	35.3	38.0	-4.7	-4.8	4.5	4.9	-4.4	2.5	-1.7	28.9
car	-6.1	91.3	-2.9	0.7	1.1	7.5	-7.4	9.3	84.5	86.3
bird	0.6	-0.4	26.0	-9.6	-2.4	22.2	-4.0	89.6	3.2	25.1
cat	-5.8	90.6	-4.5	72.5	-4.2	-2.9	-6.6	89.9	2.0	50.7
deer	6.7	5.7	-7.5	-7.7	75.1	80.1	-4.7	89.7	-3.4	3.7
dog	-4.6	91.4	-3.7	-12.3	1.7	79.5	-5.6	89.7	2.6	71.9
frog	-0.2	91.2	-5.3	-8.6	2.1	60.5	71.5	11.2	-3.9	4.5
horse	-3.2	91.0	-6.7	-12.9	14.5	79.8	-5.3	89.4	2.8	15.5
ship	4.3	90.1	-4.9	-6.7	1.3	-0.4	-1.8	26.3	84.7	16.4
truck	-6.2	91.4	-2.3	-8.1	0.0	73.5	-6.6	25.2	48.2	86.0

Table A.4: Erasing top-5 eigen directions from linear FC layers.

B Diffusion Models

B.1 Attention Visualization

We first visualize the attention layer activations to understand the importance of these layers in the diffusion process in Figure B.1, B.2. We observe that certain attention heads consistently attend to particular concepts. For example, as shown in the examples, layer 3 attends to the "bear" concept consistently while layer 12 attends to the "car".

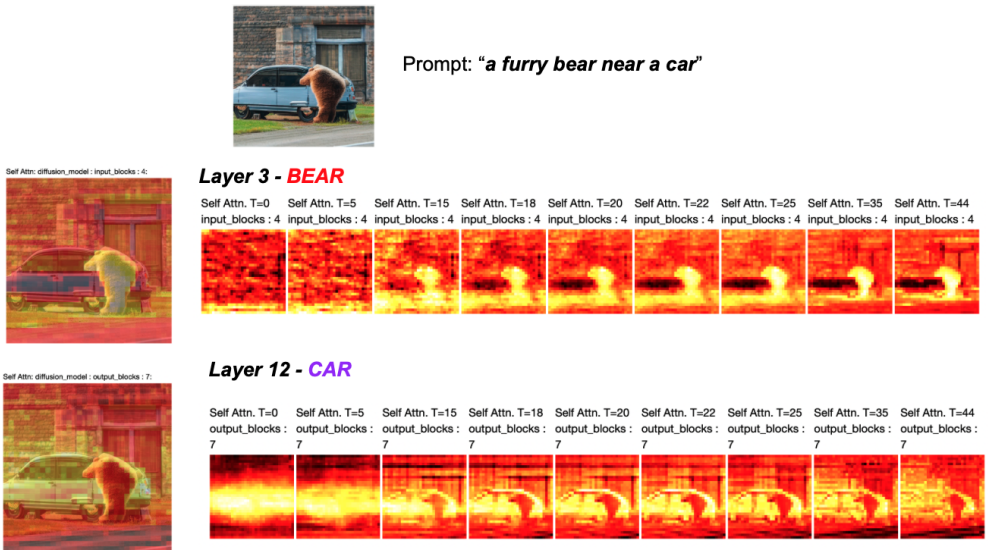

Figure B.1: Visualizing the attention heads that activate certain concepts shows that the model uses the attention mechanism to store the concepts within them. We find that layer 3 self-attention heads attend to "bear" more consistently. Therefore erasing eigen directions of "bear" activations from layer 3 could lead to erasing bear from diffusion model memory. Although it is still unsure how much other layers contribute to storing the knowledge.

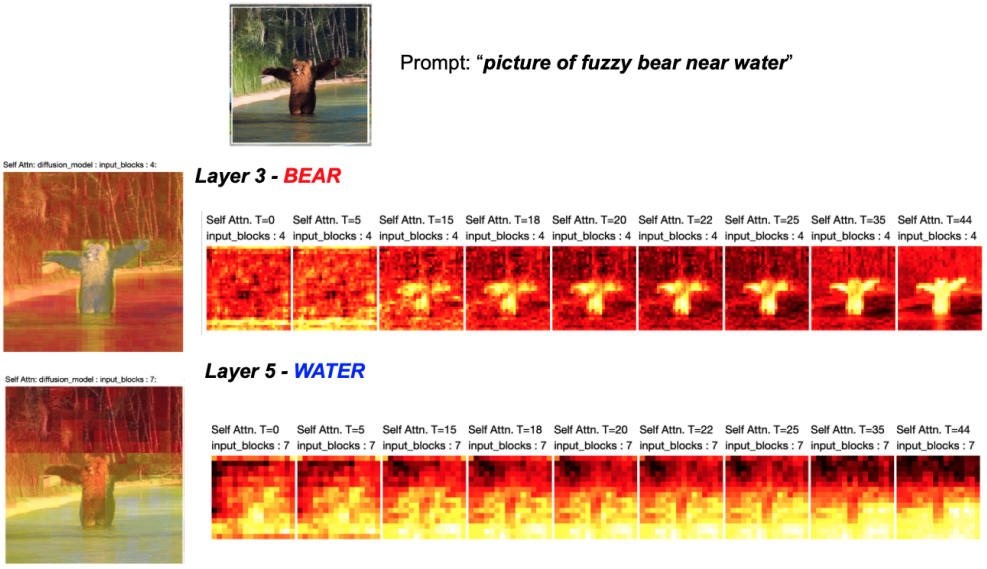

Figure B.2: We find that layer 3 self-attention heads always attend to "bear" more consistently. Therefore erasing eigen directions of "bear" activations from layer 3 could lead to erasing bear from diffusion model memory. Although it is still unsure how much other layers contribute to storing the knowledge.

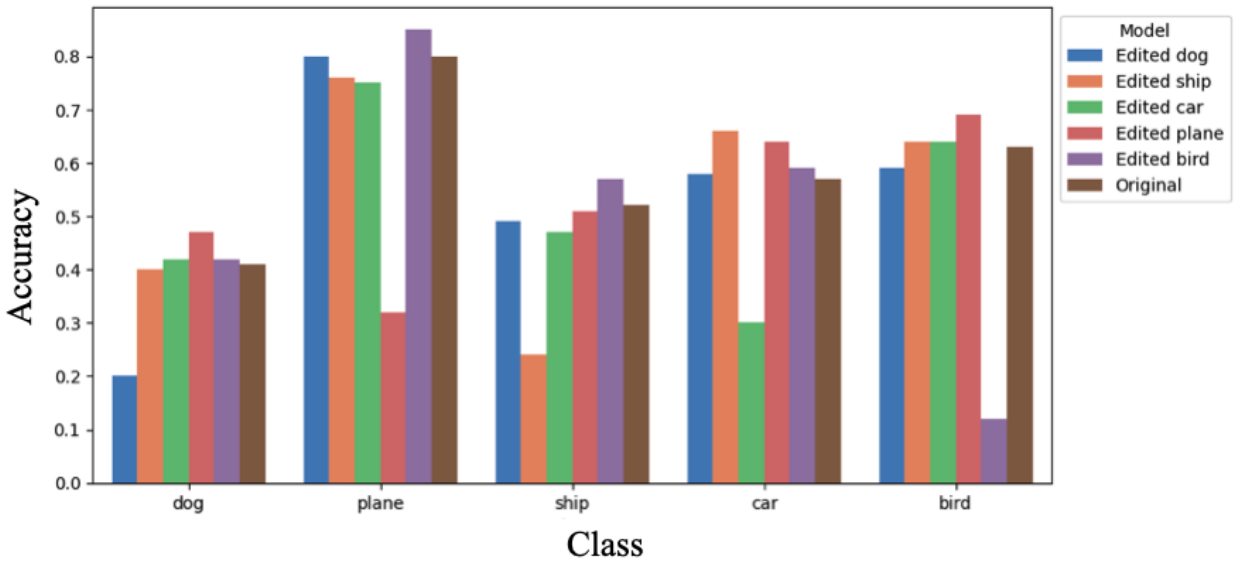

Figure B.3: Removing one concept from the diffusion model, does not affect its specificity on drawing other concepts. We find that the other class accuracies are mostly intact compared to the unedited model generation.

B.2 Model Editing

In this section, we discuss more on the effects of removing concepts by the means of finetuning as described in section 2.3. We first show the specificity retention of the model after editing a concept in Figure B.3. We generate 500 images, 100 per every 5 classes conditioned on the class name (Eg: "Picture of a dog"). We then classify the generated images using the standard resnet-50 classifier. We note the class-wise classification accuracies both original and after editing. We find that editing the model to remove one concept does not affect the model's ability to draw other concepts. We also show various concept erasure examples in Figures B.4, B.5, B.6, B.7, B.8, B.9.

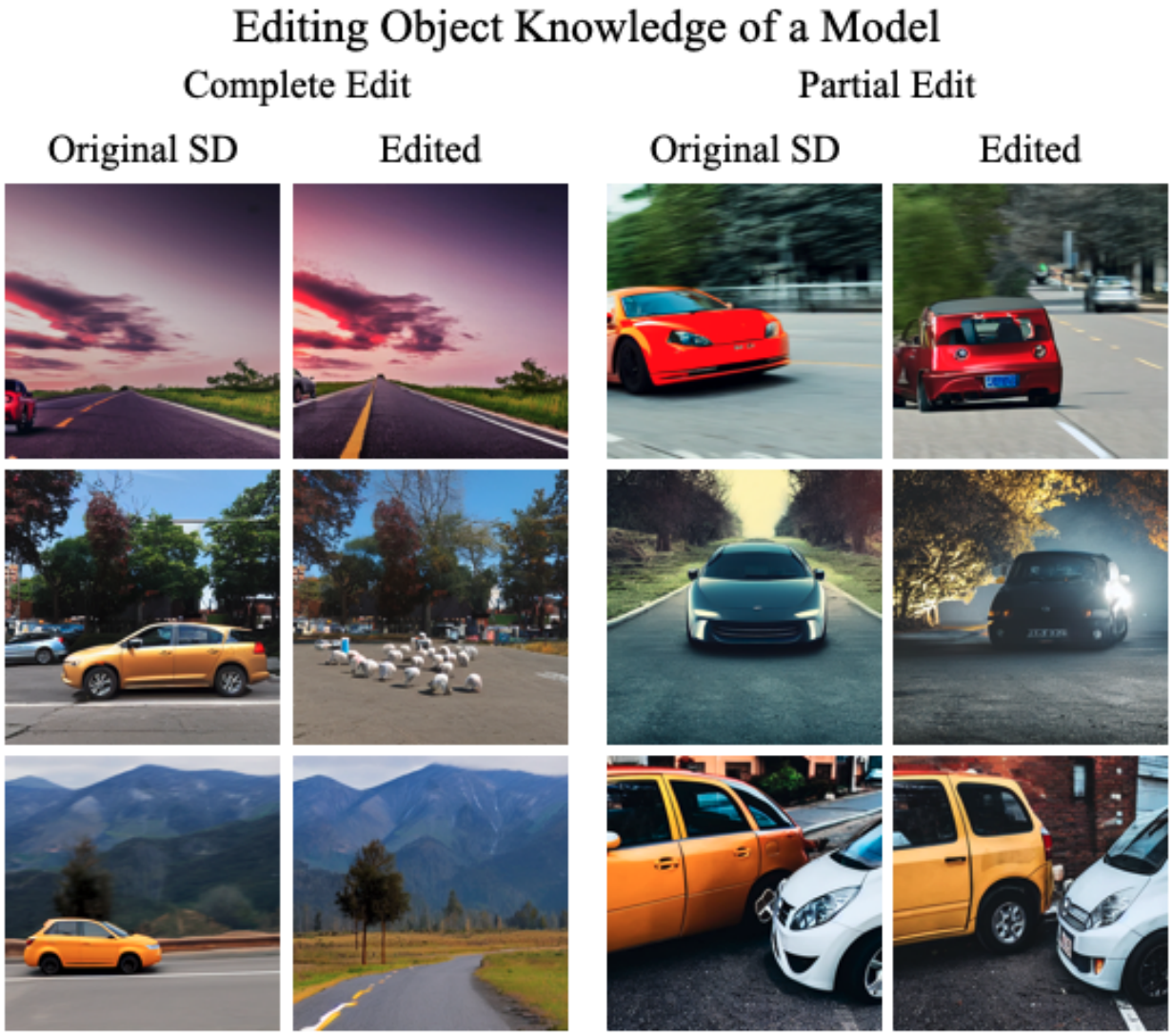

Figure B.4: Removing cars from diffusion model. We show both complete (left) and partial erasures (right).

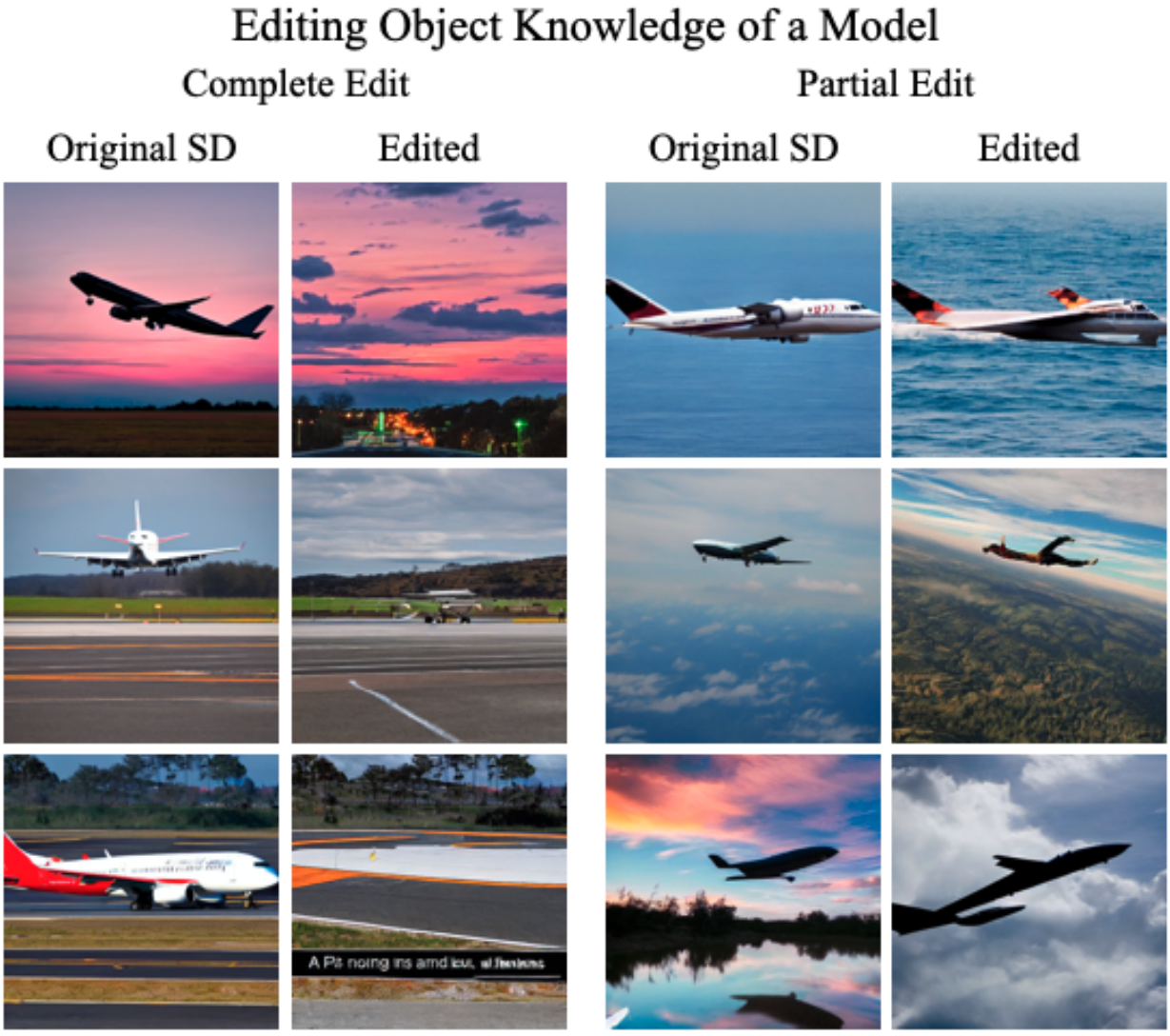

Figure B.5: Removing plane from diffusion model. We show both complete (left) and partial erasures (right).

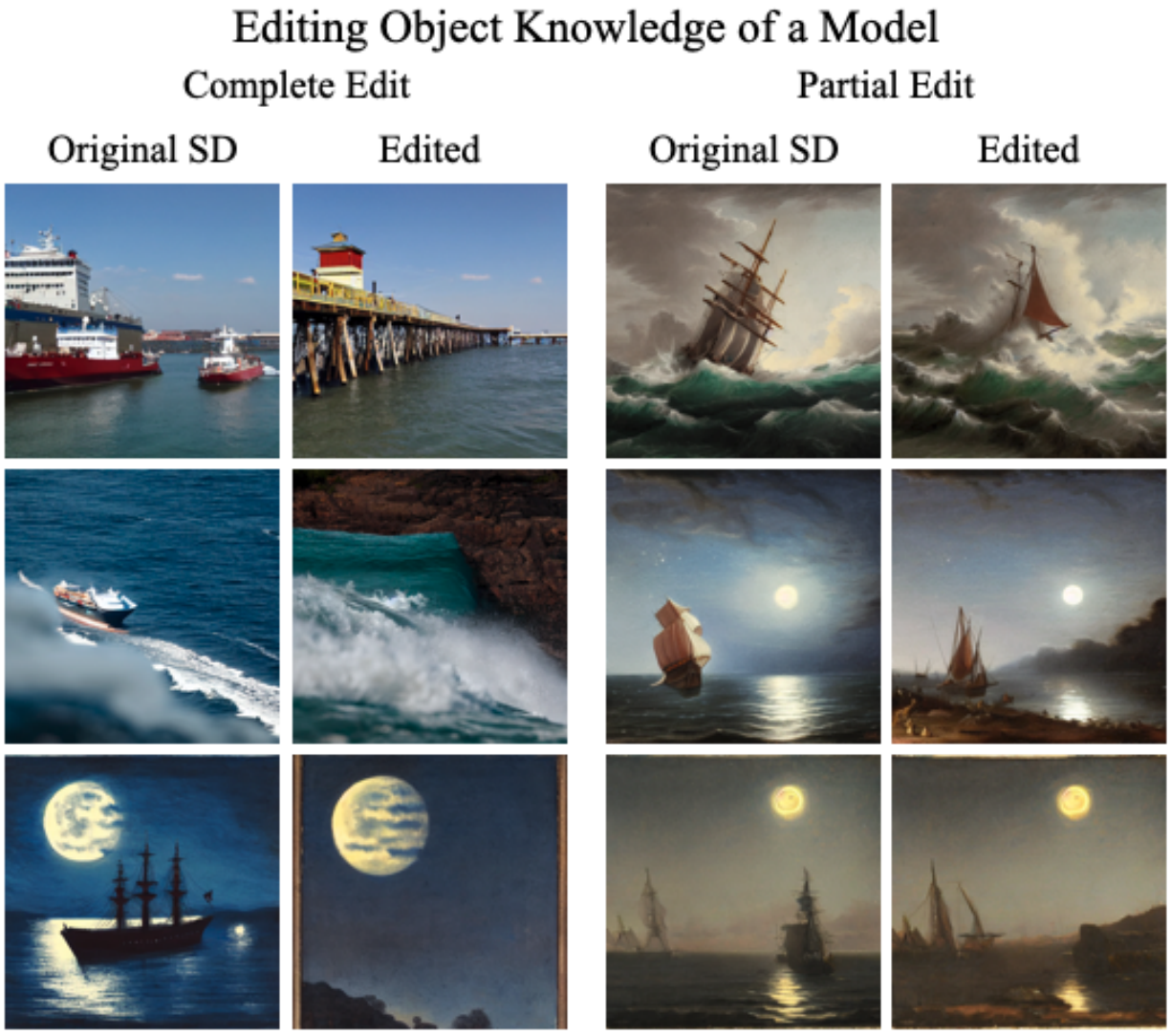

Figure B.6: Removing ship from diffusion model. We show both complete (left) and partial erasures (right).

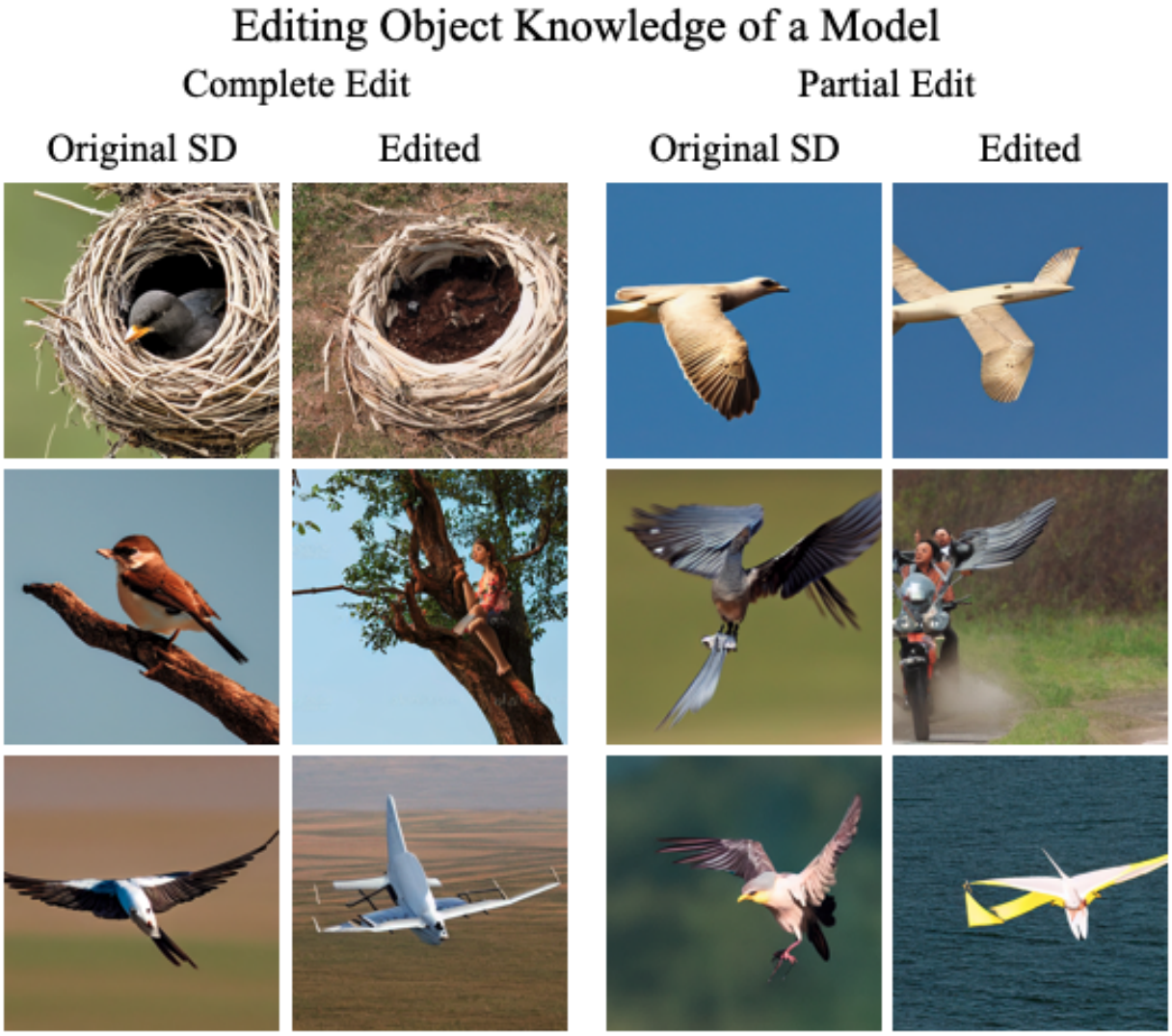

Figure B.7: Removing bird from diffusion model. We show both complete (left) and partial erasures (right).

Figure B.8: Removing dog from diffusion model. We show both complete (left) and partial erasures (right).

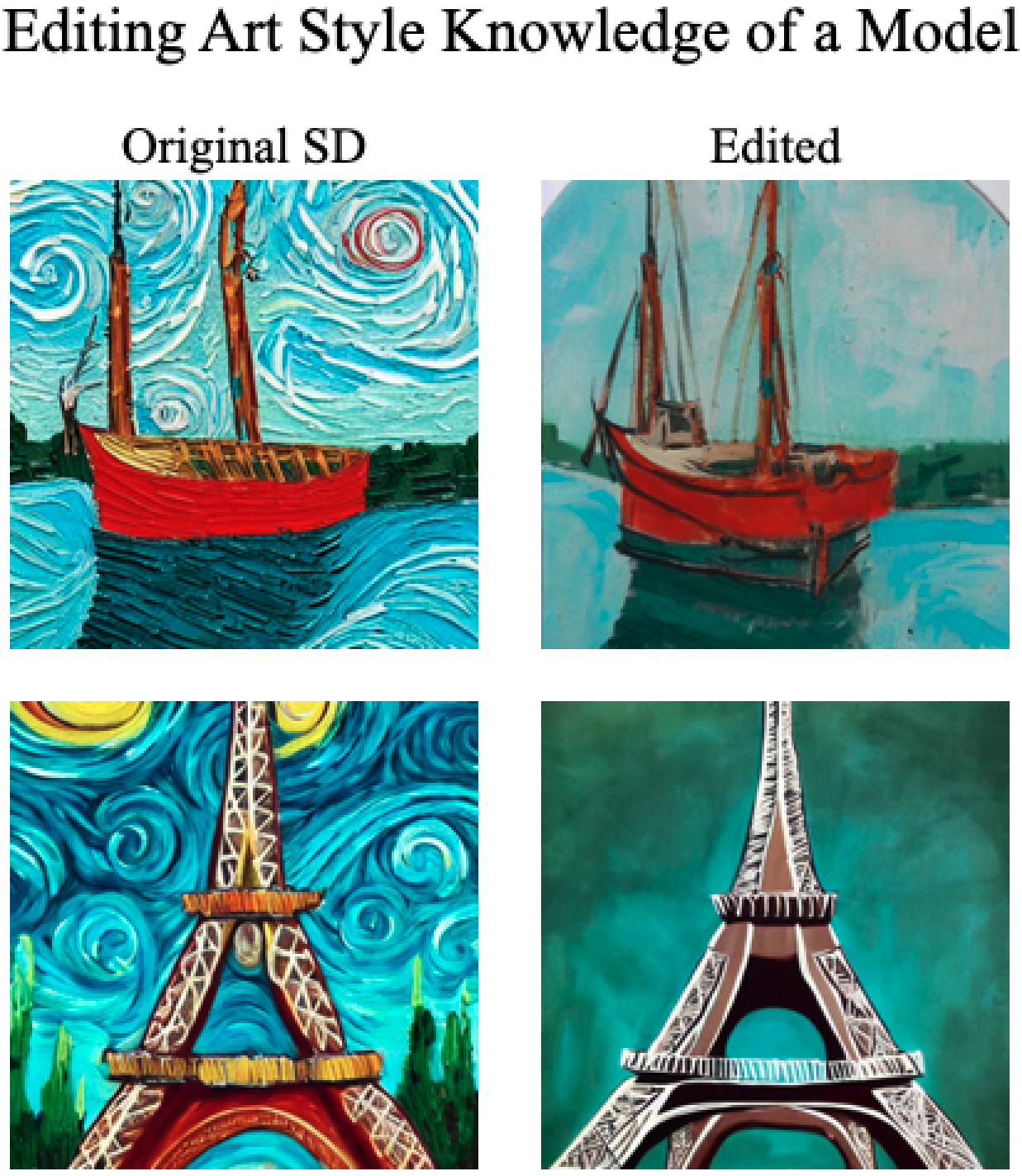

Figure B.9: Removing Van Gogh art style from diffusion model.