The Journey, Not the Destination: How Data Guides Diffusion Models

Kristian Georgiev [*1] **Joshua Vendrow** [*1] **Hadi Salman** [1] **Sung Min Park** [1] **Aleksander Mądry** [1]

Abstract

Diffusion-based generative models can synthesize photo-realistic images of remarkable quality and diversity. However, *attributing* these images back to the training data—that is, identifying specific training examples which caused an image to be generated—remains a challenge. In this paper, we propose a framework that: (i) frames data attribution in the context of diffusion models, (ii) provides a method for computing such attributions efficiently, and (iii) allows us to *counterfactually* validate them. We then apply our framework to CIFAR-10 and MS COCO datasets.

1. Introduction

Diffusion-based generative models can generate images that are simultaneously photo-realistic and highly controllable with textual prompting (Ramesh et al., 2022; Rombach et al., 2022). Beyond the architectural and algorithmic aspects, the key driver of diffusion model performance is training them on massive amounts of data (Schuhmann et al., 2022).

However, while it is clear that generated images depend on the training data, we currently lack a method for *attributing* such generated images back to the most influential training examples—that is, identifying examples which *caused* a given image to be generated.

Such attribution would enable us to tackle some of the key challenges in deploying generative models. For example, data attribution can help us diagnose various failure modes of generative models, such as bias propagation (Luccioni et al., 2023; Perera & Patel, 2023), memorization (Carlini et al., 2023), and mode collapse (Thanh-Tung & Tran, 2020). Moreover, in light of copyright concerns, data attribution can inform a principled way for assigning credit to content creators. Overall, identifying (and mitigating) these problems is critical to having more control over the generated data, especially as diffusion models are now increasingly used across the entire machine learning pipeline, including training (Azizi et al., 2023) and model evaluation (Kattakinda et al., 2022; Wiles et al., 2022; Vendrow et al., 2023). Motivated by the above, we thus ask:

How can we attribute images synthesized by diffusion models back to training data?

Indeed, data attribution has been extensively studied in the context of *supervised* learning (Koh & Liang, 2017; Ghorbani et al., 2019; Jia et al., 2019; Ilyas et al., 2022; Park et al., 2023). However, when it comes to the generative setting, data attribution poses new challenges. In particular, it is unclear *what* impact we hope to quantify with our attributions. For example, given a generated image, certain training images might be responsible for the look of the background, while other ones might be responsible for the choice of an object appearing in the foreground.

Our contributions. In this work, we present a framework for data attribution for diffusion models. In this framework, we cast data attribution as the identification of the most influential training examples at each point along a given sampling trajectory of the diffusion model. We then provide an efficient method for computing such attributions by leveraging data attribution methods developed for the supervised setting (Ilyas et al., 2022; Park et al., 2023).

Finally, we apply our method to denoising diffusion probabilistic models (DDPM) (Ho et al., 2020) trained on CIFAR-10, and latent diffusion models (LDMs) (Rombach et al., 2022) trained on MS COCO and obtain attributions that are both salient and counterfactually validated.

1.1. Related Work

While we are not aware of any prior work on attribution for diffusion models, recent works have studied *memorization*. Memorization can be thought of as a special case of data attribution where only few, nearly identical images in the training set are responsible for the generation of a corresponding image. In particular, Somepalli et al. (2022); Carlini et al. (2023) use image similarity metrics (ℓ_2 distance in pixel space and CLIP embeddings, respectively) to pinpoint cases of memorization in diffusion models. In

[*]Equal contribution [1]MIT. Correspondence to: Kristian Georgiev <krisgrg@mit.edu>.

Workshop on Challenges in Deployable Generative AI at International Conference on Machine Learning (ICML), Honolulu, Hawaii, USA. 2023.

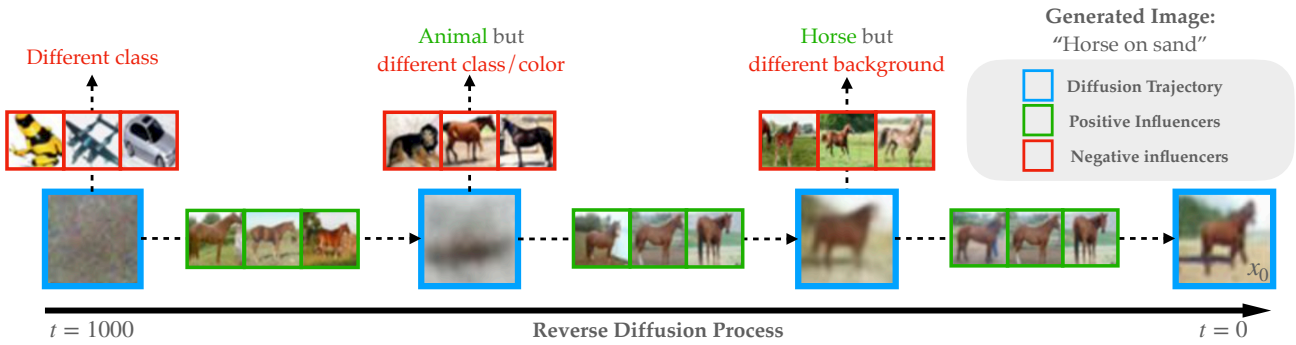


Figure 1. **Overview of our attribution method.** For a given synthesized image x_0, we apply our attribution method at individual steps (denoted by t) along the sampling trajectory. At each step, our method pinpoints the training examples with the highest influence (positive in green, negative in red) on the generative process. In particular, training examples with positive attribution scores guide the trajectory towards x_0; in contrast, ones with negative scores guide the trajectory away from x_0. For more examples, see Appendix G.

Appendix C, we present a more complete coverage of the related work.

2. Formalizing Attribution for Diffusion Models

We start by introducing background on data attribution (Section 2.1) and diffusion models (Section 2.2), and then proceed with formalizing our framework for attributing diffusion models (Section 2.3).

2.1. Data attribution

Broadly, the goal of training data attribution (Koh & Liang, 2017; Ilyas et al., 2022; Hammoudeh & Lowd, 2022) is to trace model outputs back to individual examples in the training dataset S. Intuitively, we want to estimate how the composition of the training set impacts some model output of interest (e.g., the cross-entropy loss of a classifier).

More formally, given an input z and a *model output function* $f(z;\theta)$ (where θ denotes the model parameters), a *data attribution method* is a function $\tau\colon \mathcal{X} \to \mathbb{R}$ that assigns a score $\tau(z)_i$ to each training example $z_i \in S$, indicating the change in $f(z;\theta)$ induced by removing z_i from S.

As in Ilyas et al. (2022), we adopt the perspective that a useful attribution should be *counterfactually predictive*. Concretely, if we remove datapoints T from the training set, re-train to obtain a model θ', the change in model output $f(z;\theta') - f(z;\theta)$ should correlate with the effect $\sum_{z_i \in T} \tau(z_i)$ estimated by the attribution method τ.

2.2. Diffusion models and progressive estimation

Given a data distribution $q(\cdot)$ of natural images, generative models learn a distribution $p_\theta(\cdot)$ approximating $q(\cdot)$. At a high level, diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020) sample from $p_\theta(\cdot)$ by progressively denoising an initial "noise image" $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$ for T rounds. Formally, diffusion models are a class of latent variable generative models where the latent variables typically $\mathbf{x}_T, \ldots, \mathbf{x}_1$ share the same sample space as the data. Specifically, the diffusion model ε_θ is trained to predict the noise from a noised image $\mathbf{x}_t$ given the timestep t.

At each timestep t, one can approximate the final image $\mathbf{x}_0$ using only information up until $\mathbf{x}_t$:

$$\hat{\mathbf{x}}_0^{(t)} := \left(\mathbf{x}_t - \sqrt{1-a_t}\varepsilon_\theta(\mathbf{x}_t, t)\right) / \sqrt{a_t}, \tag{1}$$

where $\{a_t\}_{t \le T}$ is a fixed monotonically decreasing sequence parameterizing the diffusion model. We refer to $\{\hat{\mathbf{x}}_0^{(T)}, \hat{\mathbf{x}}_0^{(T-1)} \ldots, \hat{\mathbf{x}}_0^{(0)} = \mathbf{x}_0\}$ as the *sampling trajectory* of the diffusion model.

2.3. Attributing the diffusion sampling trajectory

Going back to our definition of data attribution, we need to choose an appropriate model output function f for our diffusion model. One natural option is to define $f_{\text{full}}(\mathbf{x}_T;\theta) = \mathbf{x}_0$, where $\mathbf{x}_T$ is the initial random noise, and $\mathbf{x}_0$ is the final generated image. This corresponds to attributing the entire diffusion process to a given datapoint. However, computing such attribution directly is undesirable for two reasons. First, computing f_{full} involves multiple iterated calls to the diffusion model $\varepsilon_\theta(\cdot)$. Thus computing $\nabla_\theta f_{\text{full}}(\mathbf{x}_T;\theta)$ would require us to backpropagate through a large number of iterated model calls, which is numerically unstable and computationally intractable. Also, as (Ho et al., 2020) have shown, image features appear progressively throughout the sampling trajectory, e.g., the background of an image may appear early on, while the foreground may appear later. Hence, we choose instead to compute attributions at each timestep independently using a timestep-specific function $f_t(\mathbf{x}_t;\theta)$.

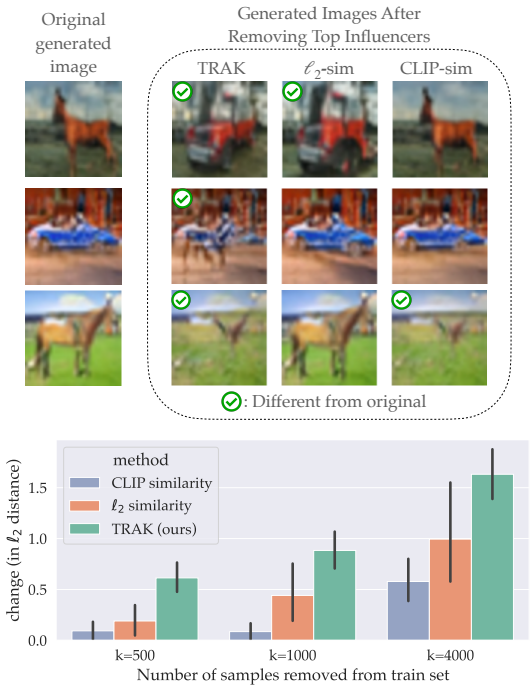


Figure 2. **Counterfactual evaluation.** We report the impact (i.e., change in ℓ_2 distance of the final generated image $\mathbf{x}_0$) induced by removing the highest scoring training examples according to our method, CLIP similarity, and ℓ_2 similarity (and re-training). **(Top)** Comparison of the original synthesized samples to those generated from the same random seed with the re-trained models. **(Bottom)** To quantify the impact of removing these images, we measure the ℓ_2 distance between 60 synthesized samples and corresponding images generated by the re-trained models when sampling from the same random seed. Error bars represent 95% confidence intervals.

3. Methods

At a high level, we want the timestep-dependent model output function f_t to capture the likelihood of the model generating $\mathbf{x}_{t-1}$ from $\mathbf{x}_t$. As a computationally cheaper proxy for that, we define each f_t to be the reconstruction loss (see (3)) of the diffusion model at timestep t.

Intuitively, we can decompose the i-th entry of an attribution vector $\tau(z)$ into the following components: (i) the change in model parameters θ when training without the i-th training example, and (ii) the induced change in model output. In particular, we can approximate the attribution function as:

$$\tau(z)_i \approx \underbrace{(\theta - \theta_{-i})}_{\text{change in model parameters}} \cdot \overbrace{\nabla_\theta f(z;\theta)}^{\text{change in model output}}, \quad (2)$$

where we denote by θ_{-i} the model parameters trained without the i-th training example.

We compute attribution scores $\tau(\cdot)$ by adapting TRAK (Park et al., 2023) and estimate each component of Equation (2).

Estimating change in parameters. At a high level, TRAK computes the change in parameters induced by removing a training point (the first term in (2)) by linearizing the model[1], and applying classical data attribution methods for generalized linear models (Pregibon, 1981). In practice, this step only requires computing per-example gradients of the training loss. See Section 3 of Park et al. (2023) for details.

Estimating gradients of model output. In classification settings, TRAK computes the change in model output (the second term in (2)) in the same way as the change in parameters. However, unlike in classification settings for which TRAK was originally designed, diffusion models are applied differently during training and inference. Specifically, during inference (generation) the model is applied *iteratively* to progressively sample an image. In particular, the model at timestep t is applied *not* to the final image $\mathbf{x}_0 := \hat{\mathbf{x}}_0^{(0)}$ (which is unavailable during sampling), but rather to an intermediate estimate $\hat{\mathbf{x}}_0^{(t)}$ of the synthesized image.

Motivated by this observation, we decompose the sampling process by attributing each individual timestep separately using the timestep-dependent model output $f_t(\mathbf{x})$, as defined in Section 2. Then, when we attribute a given timestep t, we match the sampling process along two axes. First, we treat the *predicted* final image $\hat{\mathbf{x}}_0^{(t)}$ (at timestep t) as the *actual* final image $\mathbf{x}_0$ (which, again, is unavailable at inference time). Second, we measure the reconstruction loss (i.e., how well the diffusion model is able to denoise a noisy image) when adding noise with magnitude matching the sampling process at timestep t. Specifically, we compute the Monte Carlo estimate

$$f_t\left(\hat{\mathbf{x}}_0^{(t)}\right) = \frac{1}{k}\sum_{i=1}^{k}\left\|\boldsymbol{\varepsilon}_i - \boldsymbol{\varepsilon}_\theta\left(\sqrt{\bar{\alpha}_t}\hat{\mathbf{x}}_0^{(t)} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\varepsilon}_i, t\right)\right\|_2^2 \quad (3)$$

where $\boldsymbol{\varepsilon}_i \sim \mathcal{N}(0,1)$ for all $i \in [k]$, and $\bar{\alpha}_t$ is the DDPM variance schedule (Ho et al., 2020).

Once we have the above two components, we can plug them into Equation (2) to obtain the desired attribution scores.

4. Experiments

We now evaluate our method using denoising diffusion probabilistic models (DDPMs) trained on CIFAR-10 and latent diffusion models (LDMs) trained on MS COCO (see Appendix A for setup details).

Data attribution on CIFAR-10. We use our method (see Section 3) to compute attribution scores along the sampling

[1] TRAK computes the after kernel, empirical NTK at convergence (Jacot et al., 2018; Atanasov et al., 2022; Wei et al., 2022).

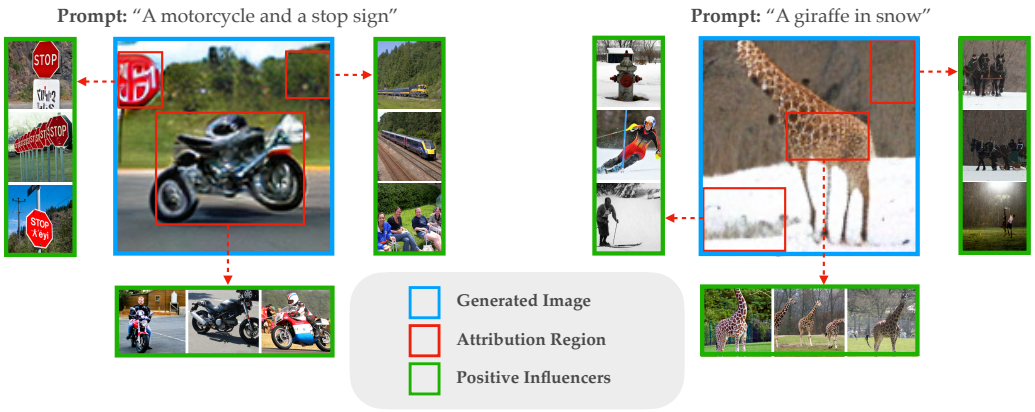


Figure 3. **Patch-based attribution.** Attributions restricted to user-specified patches (denoted with red squares) of a generated image. We show examples of attributing patches capturing individual concepts in images from a latent diffusion model trained on MS COCO.

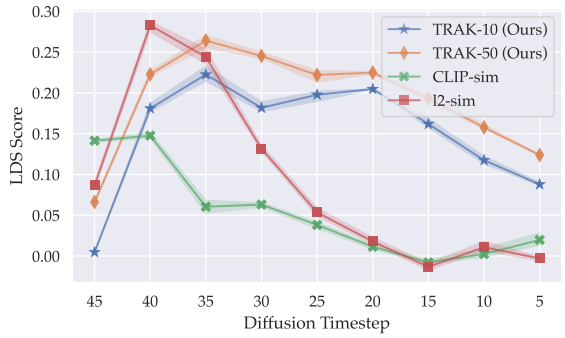


Figure 4. **Predicting model behavior.** Evaluation of the counterfactual predictiveness of different attributions using the LDS score at every fifth timestep of the sampling trajectory for TRAK (computed using 10 or 50 models), CLIP similarity, and ℓ_2 distance. Smaller timesteps are closer to the final synthesized image.

trajectory of DDPMs trained on CIFAR-10. In Figure 1, we visualize the sampling trajectory for a generated image along with the most positive and negative influencers identified by TRAK (see Appendix G for further examples). We find that positive influencers resemble the generated image throughout, while negative ones tend to differ from the generated image along specific attributes (e.g., class, background, color) depending on the corresponding timestep. Interestingly, the negative influencers increasingly resemble the generated image towards the end of the sampling trajectory. (We suspect that this is caused by the fact that the model has already generated most of the image by this point, and very dissimilar images are no longer influential).

Counterfactuals: Removing top influencers and retraining. To evaluate the faithfulness of our attribution scores, we compare the change in pixel space resulting from generating images from the same random seed using a model trained after removing the top positive influencers identified by TRAK, as well as CLIP-similarity and ℓ_2-similarity (see Figure 2). Our results suggest that TRAK is able to identify training points that have a significant impact on the sampling trajectory of the diffusion model, and outperforms in this regard the other two baselines (see Appendix F for additional counterfactual experiments).

LDS: Predicting model behavior when trained on a subset of the data. To quantify the counterfactual predictiveness of attribution scores more generally, we use the *linear datamodeling score* (LDS) (Ilyas et al., 2022), which evaluates the extent to which the attribution scores can predict the behavior of models trained on random subsets of the training dataset; see Appendix B for more details. To compute the LDS, we train additional CIFAR-10 models on random subsets of the training set. We then measure the correlation between the true and predicted outputs from TRAK, CLIP-similarity, and ℓ_2-similarity at every five time steps along the sampling trajectory (see Figure 4). Unlike in many computer vision settings (Zhang et al., 2018), we find that ℓ_2-similarity in the pixel space has competitive performance in comparison with CLIP, especially towards the beginning of the sampling trajectory. However, only TRAK remains counterfactually predictive across the entire trajectory.

Attributing Patches. Oftentimes, a synthesized image may contain multiple concepts or features. By restricting our model output function to specific patches (we apply a pixel-wise mask to Equation (3)), we can attribute only to those parts of the generated images. To test that, we generate images containing multiple concepts using textual prompts fed to an MS COCO-trained LDM. We then manually create per-concept masks for which we compute TRAK scores (see Figure 3). The resulting attributions for different masks surface training examples relevant *only* to the corresponding concepts in that region. This provides us with even more fine-grained control over the attribution process, allowing us to attribute to specific patches of interest.

References

Atanasov, A., Bordelon, B., and Pehlevan, C. Neural networks as kernel learners: The silent alignment effect. In *ICLR*, 2022.

Azizi, S., Kornblith, S., Saharia, C., Norouzi, M., and Fleet, D. J. Synthetic data from diffusion models improves imagenet classification. *arXiv preprint arXiv:2304.08466*, 2023.

Carlini, N., Hayes, J., Nasr, M., Jagielski, M., Sehwag, V., Tramer, F., Balle, B., Ippolito, D., and Wallace, E. Extracting training data from diffusion models. *arXiv preprint arXiv:2301.13188*, 2023.

Feng, Q., Guo, C., Benitez-Quiroz, F., and Martinez, A. M. When do gans replicate? on the choice of dataset size. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6701–6710, 2021.

Ghorbani, A. and Zou, J. Data shapley: Equitable valuation of data for machine learning. In *International Conference on Machine Learning (ICML)*, 2019.

Ghorbani, A., Wexler, J., Zou, J., and Kim, B. Towards automatic concept-based explanations. *arXiv preprint arXiv:1902.03129*, 2019.

Hammoudeh, Z. and Lowd, D. Training data influence analysis and estimation: A survey. In *arXiv preprint arXiv:2212.04612*, 2022.

Hampel, F. R., Ronchetti, E. M., Rousseeuw, P. J., and Stahel, W. A. *Robust statistics: the approach based on influence functions*, volume 196. John Wiley & Sons, 2011.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. In *Neural Information Processing Systems (NeurIPS)*, 2020.

Ilyas, A., Park, S. M., Engstrom, L., Leclerc, G., and Madry, A. Datamodels: Predicting predictions from training data. In *International Conference on Machine Learning (ICML)*, 2022.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In *Neural Information Processing Systems (NeurIPS)*, 2018.

Jia, R., Dao, D., Wang, B., Hubis, F. A., Hynes, N., Gürel, N. M., Li, B., Zhang, C., Song, D., and Spanos, C. J. Towards efficient data valuation based on the shapley value. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, 2019.

Karlaš, B., Dao, D., Interlandi, M., Li, B., Schelter, S., Wu, W., and Zhang, C. Data debugging with shapley importance over end-to-end machine learning pipelines. *arXiv preprint arXiv:2204.11131*, 2022.

Kattakinda, P., Levine, A., and Feizi, S. Invariant learning via diffusion dreamed distribution shifts. *arXiv preprint arXiv:2211.10370*, 2022.

Koh, P. W. and Liang, P. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, 2017.

Koh, P. W., Ang, K.-S., Teo, H. H., and Liang, P. On the accuracy of influence functions for measuring group effects. In *Neural Information Processing Systems (NeurIPS)*, 2019.

Luccioni, A. S., Akiki, C., Mitchell, M., and Jernite, Y. Stable bias: Analyzing societal representations in diffusion models. In *arXiv preprint arXiv:2303.11408*, 2023.

Nichol, A., Ramesh, A., Mishkin, P., Dariwal, P., Jang, J., and Chen, M. Dalle 2 pre-training mitigations. 2022.

Park, S. M., Georgiev, K., Ilyas, A., Leclerc, G., and Madry, A. Trak: Attributing model behavior at scale. In *Arxiv preprint arXiv:2303.14186*, 2023.

Perera, M. V. and Patel, V. M. Analyzing bias in diffusion-based face generation models. In *arXiv preprint arXiv:2305.06402*, 2023.

Pregibon, D. Logistic regression diagnostics. In *The Annals of Statistics*, 1981.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

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.

Schioppa, A., Zablotskaia, P., Vilar, D., and Sokolov, A. Scaling up influence functions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8179–8186, 2022.

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

Sohl-Dickstein, J., Weiss, E. A., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, 2015.

Somepalli, G., Singla, V., Goldblum, M., Geiping, J., and Goldstein, T. Diffusion art or digital forgery? investigating data replication in diffusion models. *arXiv preprint arXiv:2212.03860*, 2022.

Thanh-Tung, H. and Tran, T. Catastrophic forgetting and mode collapse in gans. In *2020 international joint conference on neural networks (ijcnn)*, pp. 1–10. IEEE, 2020.

van den Burg, G. and Williams, C. On memorization in probabilistic deep generative models. *Advances in Neural Information Processing Systems*, 34:27916–27928, 2021.

Vendrow, J., Jain, S., Engstrom, L., and Madry, A. Dataset interfaces: Diagnosing model failures using controllable counterfactual generation. *arXiv preprint arXiv:2302.07865*, 2023.

Wei, A., Hu, W., and Steinhardt, J. More than a toy: Random matrix models predict how real-world neural representations generalize. In *ICML*, 2022.

Wiles, O., Albuquerque, I., and Gowal, S. Discovering bugs in vision models using off-the-shelf image generation and captioning. *arXiv preprint arXiv:2208.08831*, 2022.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In *Computer Vision and Pattern Recognition (CVPR)*, 2018.

A. Experimental details

Throughout our paper, we train various diffusion models on CIFAR-10 and MS COCO.

DDPM training on CIFAR-10. We train 100 DDPMs (Ho et al., 2020) on CIFAR-10 for 200 epochs using a cosine annealing learning rate schedule that starts at 1e-4. We used the DDPM architecture that match the original implementation (Ho et al., 2020), which can be found here `https://huggingface.co/google/ddpm-cifar10-32`. At inference time we sample using a DDIM scheduler with 50 inference steps.

LDM training on MS COCO. We train 20 text-conditional latent diffusion models (LDMs) (Rombach et al., 2022) on MS COCO for 200 epochs using a cosine annealing learning rate schedule that starts at 2e-4. We used the exact CLIP and VAE used from Stable Diffusion 2, but used a custom (smaller) UNet. These models can be found here `https://huggingface.co/stabilityai/stable-diffusion-2-1`. At inference time, we sample using a DDPM scheduler with 1000 inference steps.

We will open-source our code upon publication.

B. Linear Datamodeling Score (LDS)

A useful data attribution should help us answer *counterfactual* questions of the form: can a model still generate $\mathbf{x}_0$ if we trained our instead on $S' \subset S$ from the training set?

To quantify the usefulness of an attribution method for counterfactual predictions, we use the *linear datamodeling score* (Park et al., 2023) used in evaluation of prior attribution methods. The LDS measures the degree to which the attribution can be used to predict the model output f_t that would result from training on a random subset S' (averaged over random choices of S'); a score of one indicates perfect predictions, while a score of zero indicates lack of predictiveness.

More formally, viewing the output function $f(\mathbf{x}, \theta)$ as a function $f_t(\mathbf{x}, \theta(S'))$ of the training dataset $S' \subset S$, we consider the task of predicting $f(\bar{\mathbf{x}}, S')$ given S'. This is the so-called *datamodeling* task introduced in (Ilyas et al., 2022; Park et al., 2023).

Then, consider predicting f using a linear function of the attribution scores τ:

$$\hat{f}(\mathbf{x}, S) \coloneqq \mathbf{1}_{S'} \cdot \tau(\mathbf{x})$$

where $\mathbf{1}_{S'} \in \mathbb{R}^{|S|}$ is an indicator vector encoding the subset S' and $\tau(\mathbf{x})$ is the *data attribution* vector corresponding to generated sample $\mathbf{x}$.

Given this setup, the LDS is defined as the correlation between true and predicted outputs:

$$\text{LDS}(\tau, \mathbf{x}) \coloneqq \text{Spearman-}r(\{f(\mathbf{x}, S_i), \hat{f}_\tau(\mathbf{x}, S_i)\})$$

where $S_i \subset S$ are randomly sampled subsets of the training set.

C. Related Work

Data Attribution Methods for data attribution face a trade-off of computational cost and accuracy. There is a long line of work in *influence functions* (Hampel et al., 2011; Koh & Liang, 2017; Koh et al., 2019; Schioppa et al., 2022), which are efficient to compute but perform poorly in non-convex settings (such as neural networks). Other works use Shapley values (Ghorbani & Zou, 2019; Jia et al., 2019; Karlaš et al., 2022) or data models (Ilyas et al., 2022), which more accurately perform data attribution but are significantly more computational expensive.

Memorization in Generative Models Prior to the increasing popularity of diffusion models, a number of previous works studied memorization in other generative models. For example, (Feng et al., 2021) study the impact of properties of a dataset (size, complexity) on training data replication in Generative Adversarial Networks (GANs), and (van den Burg & Williams, 2021) introduce a memorization score for Variational Autoencoders (VAEs) that can be additionally applied to arbitrary generative models. Following the release of large text-to-image diffusion models, the creators of one of these models (DALL·E 2) investigated memorization issues themselves and found that memorization could be significantly

decreased through de-duplication of the training data (Nichol et al., 2022). Recently, (Somepalli et al., 2022) explored the data replication behavior of diffusion models from the lens of 'digital forgery,' and identified many cases where, even when Stable Diffusion produced 'unique' images, it directly copies style and semantic structure from individual images in the training set. On the other hand, (Carlini et al., 2023) investigate memorization from the perspective of privacy, and show that query access to diffusion models can enable an adversary to directly extract the models' training data.

D. Diffusion models are consistent across seeds

A priori, two independent models trained on the same dataset do not share the same latent space. That is, a given noise sequence $\epsilon_T, ..., \epsilon_0$ could be denoised to two unrelated images for two different models. However, we find empirically that latent spaces from two diffusion models are highly aligned; we call this property *seed consistency*. In fact, we find that images generated by many independently trained DDPMs on CIFAR-10 from the same random seed and nearly indistinguishable (see Figure 5, right). To evaluate seed consistency quantitatively, we measure the ℓ_2 distance between images generated by two models when using identical or distinct noise sequences, and find that matching the noise sequences leads to a far smaller ℓ_2 distances (see Figure 5, left).

We additionally evaluate seed consistency on multiple checkpoints of Stable Diffusion[2] and find that images generated across these models with a fixed seed share significantly more visual similarity that those generated from independent random seeds (see Figure 6.)

We take advantage of this property when evaluating the counterfactual impact of removing the training examples relevant to a given generated image. Specifically, we now expect that re-training a model on the full training set and then sampling from the same seed should produce a highly similar image to the generated image of interest. Thus, we can evaluate the counterfactual significance of removing the training examples with the top attribution scores for a given generated image by re-training and measuring the distance (in pixel space) of an image synthesized with the same seed to the original generated image.

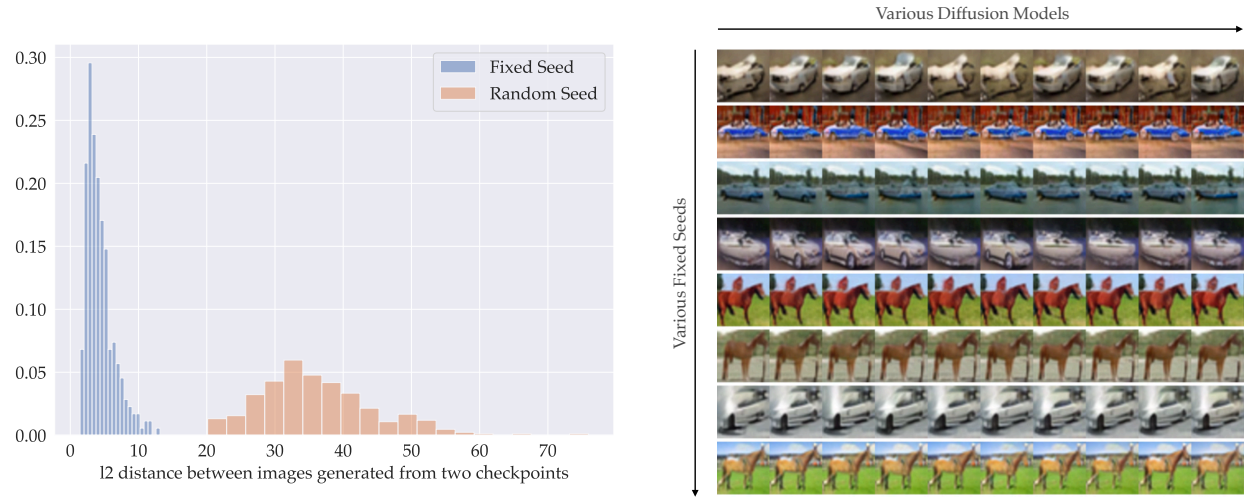


Figure 5. **Seed consistency of CIFAR-10 DDPMs**. We find that across DDPMs trained independently on CIFAR-10, when using a fixed random seed during sampling, the resulting synthesized images are very similar, and often visually indistinguishable **(Right)**. Quantitatively, we find that the ℓ_2 distance between images generated from two different models is significantly smaller when we fix the noise sequence **(Left)**.

[2]We use checkpoints provided at https://huggingface.co/CompVis/stable-diffusion and https://huggingface.co/runwayml/stable-diffusion-v1-5.

Figure 6. **Seed consistency holds for Stable Diffusion models.** We find that seed consistency holds even for large, text conditioned model, specifically for Stable Diffusion models that are trained on LAION-5B. We compare multiple checkpoints of Stable Diffusion provided by Stability AI, and find that fixing the noise sequence during sampling surfaces very similar images (in comparison to using independent noising sequences).

E. Attributing individual timesteps along the diffusion process

As additional motivation for performing attribution at individual time steps rather than the entire sampling trajectory, we highlight the following phenomena: *the same training image can be both positively influential and negative influential for a generated sample at different timesteps*. For example, consider an image of a red car on a grey background generated by our DDPM trained on CIFAR-10 (See Figure 7, top). We find that a specific training example of a red car on grass is the single most positively influential image according to TRAK at the early stages of the generative process (as it is forming the shape of the car), but is later the single most negatively influential image (possibly due to the difference in background, which could steer the model in a different direction). If we were to create a aggregate attribution score for the entire sampling trajectory, it is unclear what the attribution score would signify for this training example.

To evaluate this phenomena quantitatively, we measure the percentage of generated images for which, for a given K, there exists a training example that is one of the top K highest scoring images at some timestep and one of the top K lowest scoring images at another timestep (according to TRAK). We consider every fifth timestep, and ignore timestep 45, for which TRAK has low correlation (see Figure 4). In Figure E, we show how this percentage varies with K. As a baseline, we also include the probability of such a training example existing given completely random attribution scores. We find that our observed probabilities match those expected with random scores, signifying that an image being highly positively influential at a given timestep *does not* decrease the probability that it is highly negatively influential at a different timestep.

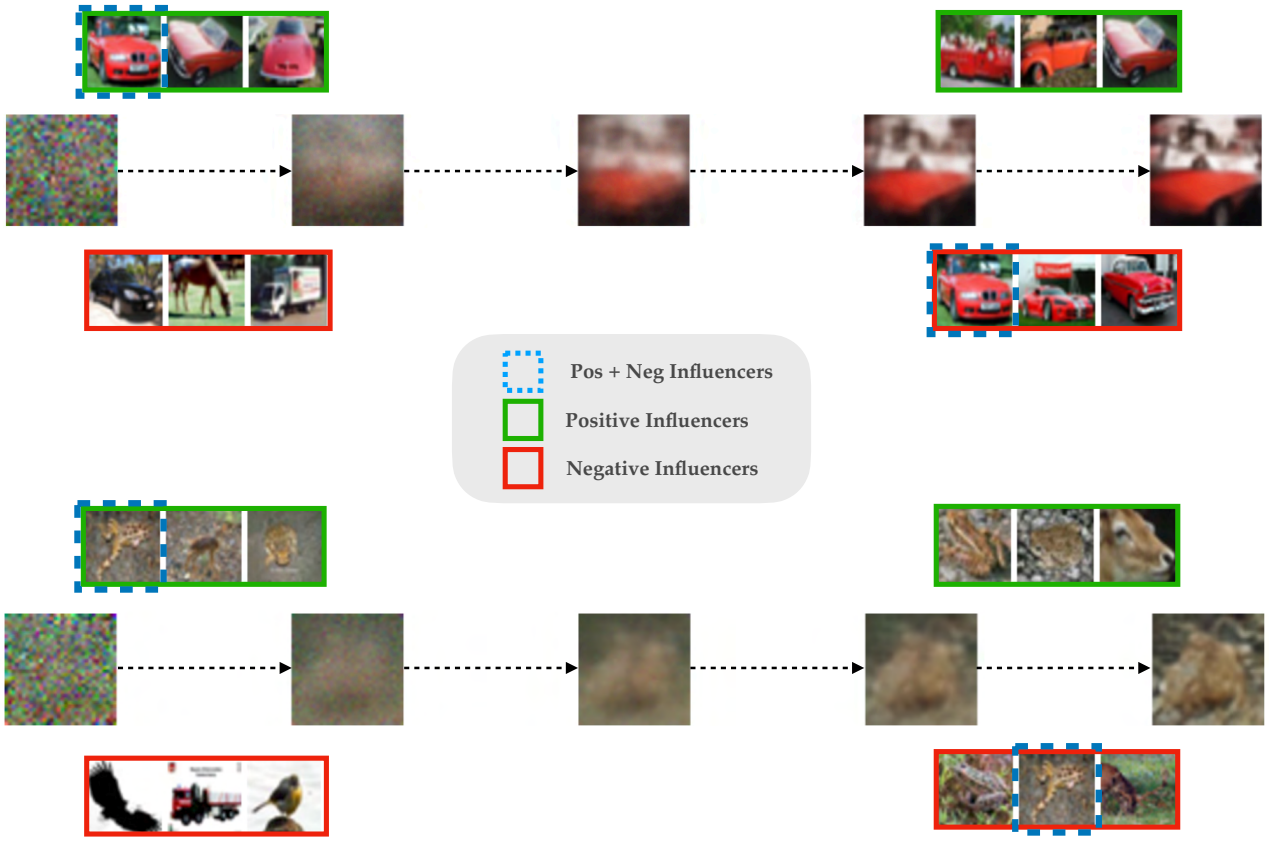


Figure 7. **Single time-step attribution reasoning.** Here, we visualize the generative process for two images generated by a DDPM on CIFAR for which there exists a training image that is both positively and negatively influential at different time steps. If we consider an aggregate attribution score across all time-steps of the sampling trajectory, we might lose the significance of such training examples which alternate between being positively and negatively influential during the sampling process.

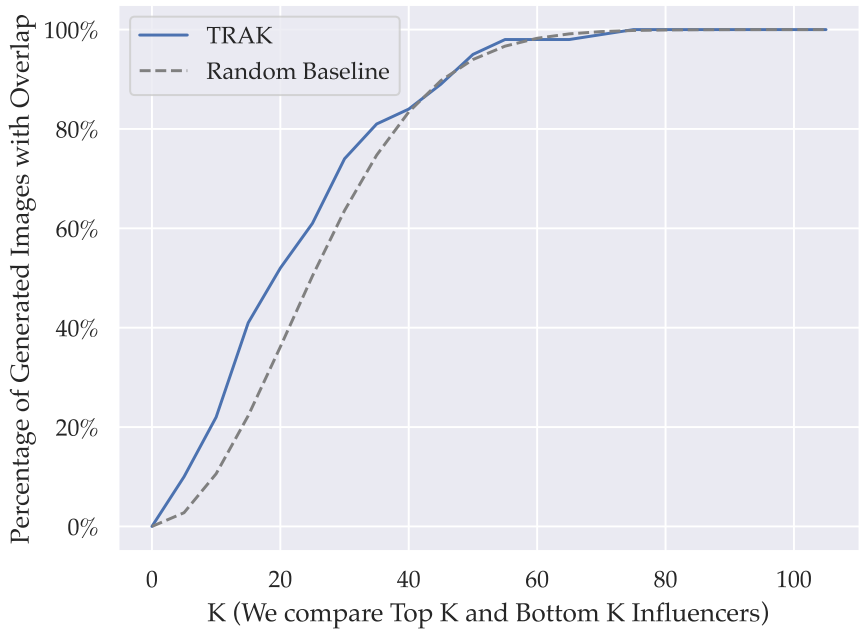


Figure 8. **The relationship between positive and negative influencers.** Here, we plot the probability that within the attribution scores for a given generated image, there exists a training example that is a one of the K most positive influencers at some timestep and one of the bottom K most negative influencers at another timestep. We compute this probability empirically with the attribution scores from TRAK and find that it closely aligns with the hypothetical baseline of completely random attribution scores. This signifies that being a top positive influencer at some timestep does not decrease the likelihood of being a top negative influencer at a different timestep.

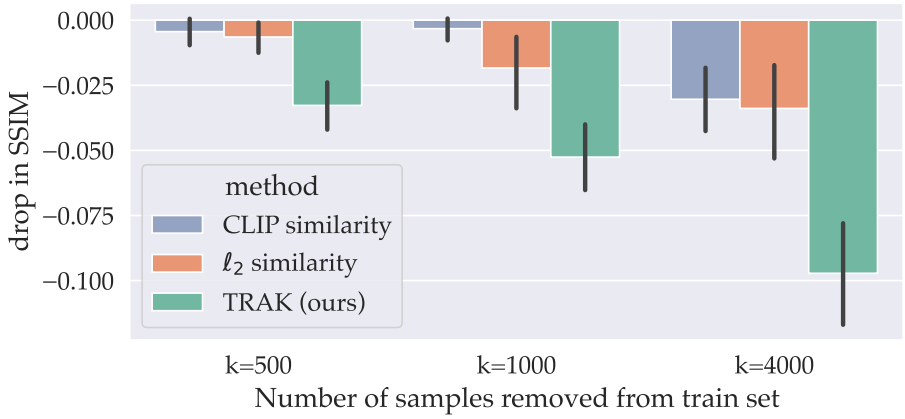
F. Omitted Counterfactual Experiments

In Figure 2 we present counterfactuals where we measure how much the generated image changes when we remove the top k most influential examples from the training set. To ground our results, we first present the change in generated images under the "null" intervention, where we simply re-train the model on the full training set. We observe that diffusion models trained independently show a remarkable consistency acorss retraining. We discuss this further in Appendix D.

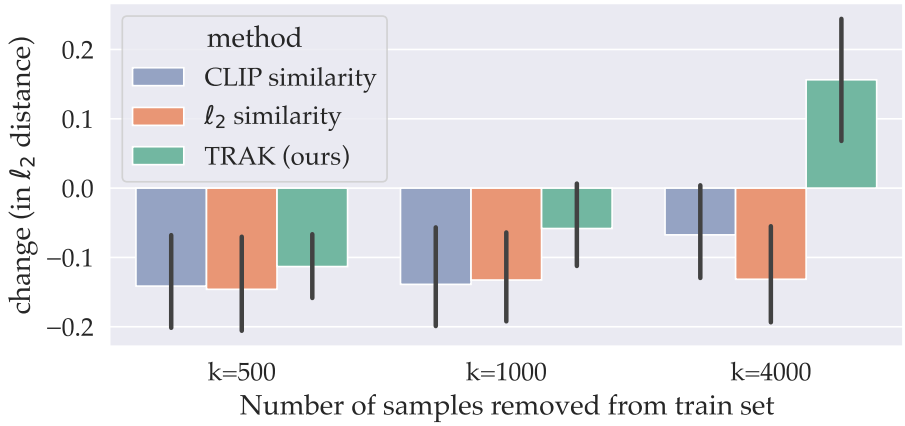
Given this consistency, we can then measure the change (in a metric of choice) in generated images before and after removing the top k most influential training examples. We train 10 models on the full CIFAR-10 training set, and one model for each removal of k examples for each of the 60 synthesized images. We then report the difference between the average distance of the image generated after re-training without the top k to the images generated by the models trained on the full data, and the average distance between the images generated by the models trained on the full data.

With our choice of model output function f_t, the TRAK attribution scores τ_t approximate the effect on *the reconstruction loss from* $\boldsymbol{x}_0^{(t)}$ *to* $\boldsymbol{x}_0^{(t-1)}$. Nevertheless, we find that this change is highly correlated with the change in various image similarity metrics (SSIM, ℓ_2 similarity) between the *final* generated images.

In addition to Figure 2, which reports the change in ℓ_2 distance in the final images, we also report the change in SSIM in Figure F.



Additionally, in Figure F we present what happens if we restart from the beginning $\mathbf{x}_0^{(t)}$ of the sampling trajectory, instead of conditioning on the trajectory up to $\mathbf{x}_0^{(t)}$.



G. Omitted plots

In this section, we present results that complement some of the result of our main paper.

G.1. More examples of the diffusion process and discovered influences.

Figure 9. Additional examples similar to Figure 1. Here, we visualize the sampling trajectory for generated images along with the most positively (green) and negatively (red) influential images at individual timesteps throughout the sampling trajectory.