Efficient Training Data Attribution on Diffusion Models

JINXU LIN

B.Eng

Supervisor: A.Prof. Chang Xu
Associate Supervisor: Dr. Daochang Liu

A thesis submitted in fulfilment of
the requirements for the degree of
Master of Philosophy

School of Computer Science
Faculty of Engineering
The University of Sydney
Australia

22 August 2025

Efficient Training Data Attribution on Diffusion Models

Jinxu Lin

Supervisor: A.Prof. Chang Xu

School of Computer Science

Faculty of Engineering

The University of Sydney

Copyright in Relation to This Thesis

Statement of Original Authorship

This is to certify that to the best of my knowledge, the content of this thesis is my own work. This thesis has not been submitted for any degree or other purposes. I certify that the intellectual content of this thesis is the product of my own work and that all the assistance received in preparing this thesis and sources have been acknowledged.

Signature:

Authorship attribution statement

I, Jinxu Lin, designed the study, conducted the data analysis, and authored the drafts. This thesis primarily builds upon the core content of the paper [1], which has been submitted to the Thirteenth International Conference on Learning Representations (ICLR 2025). I was responsible for designing and implementing the algorithm, conducting the experiments, and providing the theoretical proof for the Diffusion Attribution Score (DAS), a novel and efficient metric proposed to evaluate the influence of training data on the generation process in large-scale diffusion models.

Student Name: Jinxu Lin
Date: 30 December 2024

As supervisor for the candidature upon which this thesis is based, I can confirm that the authorship attribution statements above are correct.

Supervisor Name: Chang Xu
Date: 30 December 2024

Declaration of Use of Generative AI

I confirm that during the writing of this thesis, I utilized ChatGPT and Google Gemini as an auxiliary tool for text refinement and enhancement. The specific applications included revising sentence structure, improving fluency through paraphrasing, and performing comprehensive checks for spelling and grammatical errors. I affirm that I have thoroughly reviewed all content modified by the generative AI tool to verify its factual accuracy, integrity, and lack of bias. I maintain full intellectual responsibility for the submitted thesis, confirming that the work is original and adheres to the usage parameters outlined in the University of Sydney's generative AI guide for researchers.

Student Name: Jinxu Lin

Date: 22 August 2025

Abstract

As diffusion models gain widespread adoption, concerns over the misuse of copyrighted and private images have become increasingly prominent. A promising approach to mitigate these issues involves identifying the contribution of individual training samples in generative process, a task referred to as data attribution. Existing data attribution methods for diffusion models typically assess the contribution of a training sample by examining the change in diffusion loss when the sample is included or excluded during training. However, we contend that the direct use of diffusion loss fails to accurately capture this contribution due to the nature of its calculation. Specifically, these methods rely on computing KL-divergence, measuring the divergence between predicted and ground truth distributions. This indirect comparison of predicted distributions inadequately reflects the variations in model behavior caused by different training samples. To address these limitations, we propose the Diffusion Attribution Score (*DAS*), a novel attribution score that enables direct comparisons between predicted distributions to evaluate the importance of individual training samples. DAS is grounded in rigorous theoretical analysis, which we detail to substantiate its efficacy in attributing data influence in diffusion models. Moreover, we present optimization strategies to accelerate DAS computations, making it efficient to apply to large-scale diffusion models. Extensive experiments conducted across diverse datasets and diffusion models highlight that DAS significantly outperforms existing benchmarks, achieving superior results in terms of the linear data-modeling score and establishing a new state-of-the-art in data attribution performance.

Acknowledgements

This thesis would not have been possible without the guidance, support, and encouragement of many individuals to whom I owe my deepest gratitude. First and foremost, I would like to extend my heartfelt thanks to my supervisor, Dr. Chang Xu. At the beginning of my research journey, I often felt overwhelmed and uncertain about how to proceed. Dr. Xu provided timely guidance and steadfast support, steering me in the right direction and offering encouragement that gave me the confidence to persevere through challenging times.

I am deeply grateful to my senior colleagues, Linwei Tao and Dr. Minjing Dong, for their invaluable mentorship throughout this process. Their patient guidance helped me overcome numerous obstacles, and their meticulous feedback during the phase significantly improved the quality of my thesis. I am truly indebted for their unwavering support and generosity.

I feel privileged to have been part of a research group filled with such ta lented individuals. I would like to express my sincere appreciation to my group members: Anh-Dung Dinh, Chen Chen, Chengbin Du, Chuyang Zhou, Dr. Daochang Liu, Jianyuan Guo, Siyu Xu, Tao Huang, Xiaohuan Pei, Xiyu Wang, Yanxi Li, Yanxiang Ma, Yuemin Wu, Dr. Yunke Wang, Younan Zhu, Zijian Wang, and Zunzhi You. Your collaboration, camaraderie, and friendship have made this journey not only productive but also deeply fulfilling and enjoyable.

Finally, I wish to express my deepest gratitude to my family. To my mother, Guangying Bai, and my father, Jinchun Lin, your unwavering support and belief in me have been my foundation and source of strength throughout this journey. To my sister, Jinhui Lin, her husband, Lixiong Wen, and their son, Zhuohao Wen, thank you for always being there for me and offering encouragement when I needed it most. Your love and support have been my anchor, and I am forever grateful for all you have done.

To all of you, thank you for your invaluable contributions to this achievement. This thesis is as much a testament to your support as it is to my efforts.

Contents

CHAPTER 1

Introduction

Diffusion models, highlighted in key studies [2, 3], are advancing significantly in generative machine learning with broad applications from image generation to artistic creation [4, 5, 6, 7]. As these models, exemplified by projects like Stable Diffusion [8], become increasingly capable of producing high-quality, varied outputs, the misuse of copyrighted and private images has become a significant concern. A key strategy to address this issue is identifying the contributions of training samples in generative models by evaluating their influence on the generation, a task known as data attribution. Data attribution in machine learning is to trace model outputs back to influential training examples, which is essential to understand how specific data points affect model behaviors. In practical applications, data attribution spans various domains, including explaining predictions [9, 10, 11], curating datasets [12, 13, 14], and dissecting the mechanisms of generative models like Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs) [15, 16], serving to enhance model transparency and explore the effect of training data on model behaviors.

Data attribution methods generally fall into two categories. The first, based on sampling [17, 18, 11], involves retraining models to assess how outputs change with the a data deletion. While effective, this method requires training thousands of models. The second approach uses approximations to assess the change in output function for efficiency [9, 19, 20] by proposing attribution score function. TRAK (Tracing with the Randomly-projected After Kernel) [21] introduced a innovative estimator, which considers the inverse-sigmoid function as model output and computes attribution scores to assess its change. Building on this, [22] proposed D-TRAK, adapting TRAK to diffusion models by following TRAK's derivation and replacing the inverse-sigmoid function with the diffusion loss. Moreover, D-TRAK reported

counterintuitive findings that the output function could be replaced with alternative functions without altering the score's form. These empirical designs outperformed theoretically motivated diffusion losses in experiments, emphasizing the need for a deeper understanding of the attribution properties on diffusion models.

In this paper, we define the objective of data attribution in diffusion models as evaluating the impact of training samples on generation by measuring shifts in the predicted distribution after removing specific samples and retraining the model. From this perspective, directly applying TRAK to diffusion models by replacing the output function with the diffusion loss leads to indirect comparisons between predicted distributions, as the diffusion loss represents the KL-divergence between predicted and ground truth distributions [2]. We also address the counter-intuitive findings in D-TRAK, which manually removes the effect of data distributions in attribution but treats the diffusion model's output as a scalar, failing to capture the unique characteristics of diffusion models. Our analysis indicates that TRAK's derivation cannot be directly extended to diffusion models, as they fundamentally differ from discriminative models. To address this, we propose the Diffusion Attribution Score (*DAS*), a novel metric designed specifically for diffusion models to quantify the impact of training samples by measuring the KL-divergence between predicted distributions when a sample is included or excluded from the training set. DAS computes this divergence through changes in the noise predictor's output, which, by linearization, can be represented as variations in the model parameters. These parameter changes are then measured using Newton's Method. Since directly computing the full form of DAS is computationally expensive, we propose several techniques, such as compressing models and datasets, to accelerate computation. These enhancements enable the application of DAS to large-scale diffusion models, significantly improving its practicality. Our experiments across ranges of datasets and diffusion models demonstrate that DAS significantly outperforms previous benchmarks, including D-TRAK and TRAK, achieving superior results in terms of the linear data-modeling score. This consistent performance across diverse settings highlights its robustness and establishes DAS as the new state-of-the-art method for data attribution in diffusion models. The primary contributions of our work are summarized as follows:

(1) We provide a comprehensive analysis of the limitations of directly applying TRAK to diffusion models and evaluate D-TRAK's empirical design, highlighting the need for more effective attribution methods tailored to diffusion models.

(2) We introduce DAS, a theoretically solid metric designed to directly quantify discrepancies in model outputs, supported by detailed derivations. We also discuss various techniques, such as compressing models or datasets, to accelerate the computation of DAS, facilitating its efficient implementation.

(3) DAS demonstrates state-of-the-art performance across multiple benchmarks, notably excelling in linear datamodeling scores.

CHAPTER 2

Literature review

2.1 Diffusion Models

As deep learning advances, a variety of generative models have been introduced to synthesize realistic data, including Generative Adversarial Networks (GANs)[23], Variational Autoencoders (VAEs)[24], and Generative Pretrained Transformers (GPTs)[25]. Among these, diffusion models[2] have emerged as a powerful framework for high-quality image generation tasks [26], attracting significant attention for their ability to produce diverse and highly detailed outputs. Notable examples of discrete-time diffusion models include Denoising Diffusion Probabilistic Models (DDPMs)[2] and Latent Diffusion Models (LDMs), the latter serving as the foundation of Stable Diffusion[8].

We present an introduction to diffusion models by elaborating on the principles of Denoising Diffusion Probabilistic Models (DDPMs). Consider a training set $\mathbb{S} = \{\boldsymbol{z}^{(1)}, ..., \boldsymbol{z}^{(n)}\}$ where each training sample $\boldsymbol{z}^{(i)} := (\boldsymbol{x}^{(i)}, y^{(i)}) \sim \mathcal{Z}$ is an input-label pair[1]. Given an input data distribution $q(\boldsymbol{x})$, DDPMs aim to predict a distribution $p_\theta(\boldsymbol{x})$ to approximate $q(\boldsymbol{x})$. The predicting process is divided into forward and reverse process, conducted over a series of timesteps in the latent variable space, with $\boldsymbol{x}_0$ denoting the initial image and $\boldsymbol{x}_t$ denoting the latent variables at timestep $t \in [1, T]$. In the forward process, DDPMs sample an observation $\boldsymbol{x}_0$ from $\mathbb{S}$ and add noise on it across T timesteps: $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) := \mathcal{N}(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I})$, where $\beta_1, ..., \beta_T$ constitute a variance schedule. As indicated in DDPMs, the latent variable

[1] In text-to-image task, $\boldsymbol{z}^{(i)} := (\boldsymbol{x}^i, y^i)$ represents an image-caption sample, whereas in unconditional generation, it solely contains an image $\boldsymbol{z}^{(i)} := (\boldsymbol{x}^{(i)})$.

$\boldsymbol{x}_t$ can be express as a linear combination of $\boldsymbol{x}_0$:

$$\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \tag{2.1}$$

where $\alpha_t := 1 - \beta_t, \bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$. In the reward process, DDPMs predict a distribution $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ by minimizing the KL-divergence from data at t:

$$D_{\mathrm{KL}}[p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)\|q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)] = \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\boldsymbol{I})}[\frac{\beta_t^2}{2\alpha_t(1-\bar{\alpha}_t)}||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)||^2], \tag{2.2}$$

where $\boldsymbol{\epsilon}_\theta$ is a function implemented by models θ which can be seen as a noise predictor. A simplified version of objective function for a data point $\boldsymbol{x}$ used to train DDPMs is:

$$\mathcal{L}_{\text{Simple}}(\boldsymbol{x}, \theta) = \mathbb{E}_{\boldsymbol{\epsilon},t}[||\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t) - \boldsymbol{\epsilon}||^2]. \tag{2.3}$$

2.2 Data Attribution

The training data plays a pivotal role in shaping the behavior of machine learning models, and data attribution seeks to quantify the importance of individual training samples with respect to the desired model outputs. This task is typically addressed by considering a counterfactual question: if a training sample $\boldsymbol{z}^{(i)}$ is removed from the dataset $\mathbb{S}$, and a model $\theta_{\setminus i}$ is retrained on the subset $\mathbb{S}_{\setminus i}$, the influence of $\boldsymbol{z}^{(i)}$ on the output of a test sample $\boldsymbol{z}$ can be measured by the change in the model output, expressed as $f(\boldsymbol{z}, \theta) - f(\boldsymbol{z}, \theta_{\setminus i})$. Here, $f(\boldsymbol{z}, \theta)$ represents the model output, which could take various forms, such as the direct model prediction or the loss function.

Sampling-based approaches, such as empirical influence functions [19], Shapley value estimators [18, 27], and datamodels [11], estimate changes in model output by sampling subsets of data and retraining models on them. While conceptually straightforward and capable of providing precise assessments of training data influence, these methods are computationally prohibitive, often requiring the retraining of thousands of models, which makes them impractical for large-scale datasets. In contrast, methods like influence approximation [9, 28] and gradient agreement scoring [20] significantly reduce computational overhead by leveraging model gradients but may exhibit reduced reliability in non-convex settings [29, 30]. These

gradient-based approaches compute attribution scores that approximate changes in the model's output using gradient information, bypassing the need for retraining after removing samples and substantially lowering computational requirements. To illustrate the principles behind gradient-based attribution methods, we take TRAK [21] as a representative example.

TRAK stands out as a representative data attribution method designed for large-scale models focused on discriminative tasks. TRAK defines the model output function as:

$$f_{\text{TRAK}}(\boldsymbol{z}, \theta) = \log[\hat{p}(\boldsymbol{x}, \theta)/(1 - \hat{p}(\boldsymbol{x}, \theta))], \tag{2.4}$$

where $\hat{p}(\boldsymbol{x}, \theta)$ is the corresponding class probability of $\boldsymbol{z}$. TRAK then introduces attribution score $\tau_{\text{TRAK}}(\boldsymbol{z}, \mathbb{S})^{(i)}$ to approximate the change in f_{TRAK} after a data intervention, which is expressed as:

$$\tau_{\text{TRAK}}(\boldsymbol{z}, \mathbb{S})^{(i)} := \phi(\boldsymbol{z})^\top (\boldsymbol{\Phi}^\top \boldsymbol{\Phi})^{-1} \phi(\boldsymbol{z}^{(i)}) r^{(i)} \approx f_{\text{TRAK}}(\boldsymbol{z}, \theta) - f_{\text{TRAK}}(\boldsymbol{z}, \theta_{\setminus i}), \tag{2.5}$$

where $r^{(i)} := [1 - \hat{p}(\boldsymbol{x}^{(i)}, \theta)]$ denotes the residual for sample $\boldsymbol{z}^{(i)}$. Here, $\phi(\boldsymbol{z}) := \boldsymbol{P}^\top \nabla_\theta f_{\text{TRAK}}(\boldsymbol{z}, \theta)$ and $\boldsymbol{\Phi} := [\phi(\boldsymbol{z}^{(1)}), ..., \phi(\boldsymbol{z}^{(n)})]$ represents the matrix of stacked gradients on $\mathbb{S}$. $\boldsymbol{P} \sim \mathcal{N}(0, 1)^{d \times k}$ is a random projection matrix [31] to reduce the gradient dimension.

2.3 Data Attribution in Generative Models

The discussed methods address counterfactual questions within the context of discriminative models, focusing primarily on accuracy and model predictions. Extending these methodologies to generative models presents complexities due to the lack of clear labels or definitive ground truth. Research in this area includes efforts to compute influence within Generative Adversarial Networks (GANs) [16] and Variational Autoencoders (VAEs) [15]. In the realm of diffusion models, earlier research [32] has explored influence computation by employing ensembles that necessitate training multiple models on varied subsets of training data—a method less suited for traditionally trained models. [33] suggests an alternative termed "customization" which involves adapting or tuning a pretrained text-to-image model through a

specially designed training procedure. MONTAGE [34] integrates a novel technique to monitor generations throughout the training via internal model representations. In this paper, we mainly focus on post-hoc data attribution method, which entails applying attribution methods after training. Journey TRAK [35] extends TRAK to diffusion models by attributing influence across individual denoising timesteps. In DataInf [36], influence function using the loss gradient and Hessian have been improved for greater accuracy and efficiency in attributing diffusion models. Moreover, to adapt TRAK in diffusion models, D-TRAK [22] first followed TRAK's guidence, replacing the output function with Simple Loss: $f_{\text{D-TRAK}}(\boldsymbol{z}, \theta) = \mathcal{L}_{\text{Simple}}(\boldsymbol{x}, \theta)$, and simplifies the residual term to an identity matrix $\boldsymbol{I}$. The attribution function in D-TRAK is defined as follows:

$$\tau_{\text{D-TRAK}}(\boldsymbol{z}, \mathbb{S})^{(i)} = \phi(\boldsymbol{z})^\top (\boldsymbol{\Phi}^\top \boldsymbol{\Phi})^{-1} \phi(\boldsymbol{z}^{(i)}) \boldsymbol{I} \approx f_{\text{D-TRAK}}(\boldsymbol{z}, \theta) - f_{\text{D-TRAK}}(\boldsymbol{z}, \theta_{\setminus i}), \tag{2.6}$$

where $\phi(\boldsymbol{z}) := \boldsymbol{P}^\top \nabla_\theta f_{\text{D-TRAK}}(\boldsymbol{z}, \theta)$. Interestingly, D-TRAK observed that substituting Simple Loss with other functions can yield superior attribution performance. Examples of these include $\mathcal{L}_{\text{Square}}(\boldsymbol{z}, \theta) = \mathbb{E}_{t,\boldsymbol{\epsilon}}[||\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)||^2]$ and $\mathcal{L}_{\text{Average}}(\boldsymbol{z}, \theta) = \mathbb{E}_{t,\boldsymbol{\epsilon}}[\text{Avg}(\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t))]$. The success of the empirical designs in D-TRAK highlight the imperative for further exploration into data attribution within diffusion models.

2.4 Application of Data Attribution

Recent research has underscored the effectiveness of data attribution methods in a variety of applications. These include explaining model predictions [9, 11], debugging model behaviors [37], assessing the contributions of training data [18, 27], identifying poisoned or mislabeled data [38], most influential subset selection [39] and managing data curation [12, 14, 13]. Additionally, the adoption of diffusion models in creative industries, as exemplified by Stable Diffusion and its variants, has grown significantly [8, 40]. This trend highlights the critical need for fair attribution methods that appropriately acknowledge and compensate artists whose works are utilized in training these models. Such methods are also crucial for addressing related legal and privacy concerns [41, 42].

CHAPTER 3

Methods

3.1 Rethinking output function in diffusion models

For the goal of data attribution within diffusion models, we want to measure the influence of a specific training sample $\boldsymbol{z}^{(i)}$ on a generated sample $\boldsymbol{z}^{\text{gen}}$. This task can be approached by addressing the counterfactual question: How would $\boldsymbol{z}^{\text{gen}}$ change if we removed $\boldsymbol{z}^{(i)}$ from $\mathbb{S}$ and retrained the model on the subset $\mathbb{S}_{\setminus i}$? This change can be evaluated by the distance between predicted distribution $p_\theta(\boldsymbol{x}^{\text{gen}})$ and $p_{\theta_{\setminus i}}(\boldsymbol{x}^{\text{gen}})$ in DDPM, where the subscript denotes the deletion of $\boldsymbol{z}^{(i)}$.

Reviewing D-TRAK, they propose using $\tau_{\text{D-TRAK}}$ to approximate this difference on the Simple Loss by setting output function as $f_{\text{D-TRAK}} = \mathcal{L}_{\text{Simple}}$. In details, the difference is computed as:

$$\begin{aligned}\tau_{\text{D-TRAK}}(\boldsymbol{z}^{\text{gen}}, \mathbb{S})^{(i)} &\approx f_{\text{D-TRAK}}(\boldsymbol{z}^{\text{gen}}, \theta) - f_{\text{D-TRAK}}(\boldsymbol{z}^{\text{gen}}, \theta_{\setminus i}) \\ &= \mathbb{E}_{\boldsymbol{\epsilon},t}[||\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t^{\text{gen}}, t) - \boldsymbol{\epsilon}||^2] - \mathbb{E}_{\boldsymbol{\epsilon},t}[||\boldsymbol{\epsilon}_{\theta_{\setminus i}}(\boldsymbol{x}_t^{\text{gen}}, t) - \boldsymbol{\epsilon}||^2]. \end{aligned} \tag{3.1}$$

However, from the distribution perspective, this approach conducts an indirect comparison of KL-divergences between the predicted distributions by involving the data distribution $q(\boldsymbol{x}^{\text{gen}})$:

$$\begin{aligned}\tau_{\text{D-TRAK}}(\boldsymbol{z}^{\text{gen}}, \mathbb{S})^{(i)} &\approx D_{\text{KL}}[p_\theta(\boldsymbol{x}^{\text{gen}}) \| q(\boldsymbol{x}^{\text{gen}})] - D_{\text{KL}}[p_{\theta_{\setminus i}}(\boldsymbol{x}^{\text{gen}}) \| q(\boldsymbol{x}^{\text{gen}})] \\ &= D_{\text{KL}}[p_\theta(\boldsymbol{x}^{\text{gen}}) \| p_{\theta_{\setminus i}}(\boldsymbol{x}^{\text{gen}})] - \int [p_\theta(\boldsymbol{x}^{\text{gen}}) - p_{\theta_{\setminus i}}(\boldsymbol{x}^{\text{gen}})] \log q(\boldsymbol{x}^{\text{gen}}) \mathrm{d}x. \end{aligned} \tag{3.2}$$

Compared to directly computing the KL-divergences between predicted distributions p_θ and $p_{\theta_{\setminus i}}$, D-TRAK involves a Cross Entropy between the predicted distribution shift $p_\theta - p_{\theta_{\setminus i}}$

and data distribution q, where the crossing term is generally nonzero unless are identical distributions. This setting involves the influence of the data distribution q in the attribution process and may introduce errors when evaluating the differences. We can consider the irrationality of this setting through a practical example: the predicted distributions might approach the data distribution $q(\boldsymbol{x}^{\text{gen}})$ from different directions while training, yet exhibit similar distances. In this case, D-TRAK captures only minimal loss changes, failing to reflect the true distance between the predicted distributions.

To isolate the effect of the removed data on the model, we propose the Diffusion Attribution Score (DAS), conducting a direct comparison between p_θ and $p_{\theta_{\setminus i}}$ by assessing their KL-divergence:

$$\begin{aligned}\tau_{\text{DAS}}(\boldsymbol{z}^{\text{gen}}, \mathbb{S})^{(i)} &\approx D_{\text{KL}}[p_\theta(\boldsymbol{x}^{\text{gen}})||p_{\theta_{\setminus i}}(\boldsymbol{x}^{\text{gen}})] \\ &\approx \mathbb{E}_{\boldsymbol{\epsilon},t}[||\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t^{\text{gen}}, t) - \boldsymbol{\epsilon}_{\theta_{\setminus i}}(\boldsymbol{x}_t^{\text{gen}}, t)||^2]. \end{aligned} \tag{3.3}$$

The output function in DAS is defined as $f_{\text{DAS}}(\boldsymbol{z}, \theta) = \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t^{\text{gen}}, t)$, which is able to directly reflect the differences between the noise predictors of the original and the retrained models. Eq. 3.3 also validates the effectiveness of employing $\mathcal{L}_{\text{square}}$ as the output function, which is formulated as:

$$\tau_{\text{Square}}(\boldsymbol{z}^{\text{gen}}, \mathbb{S})^{(i)} \approx \mathbb{E}_{\boldsymbol{\epsilon},t}[||\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t^{\text{gen}}, t)||^2] - \mathbb{E}_{\boldsymbol{\epsilon},t}[||\boldsymbol{\epsilon}_{\theta_{\setminus i}}(\boldsymbol{x}_t^{\text{gen}}, t)||^2]. \tag{3.4}$$

The effectiveness of τ_{Square} lies in manually eliminating the influence of q; however, this approach has its limitations since the output of diffusion model is a high dimensional feature. Defining the output function as $\mathcal{L}_{\text{Square}}$ or using average and L^2 norm, treats the latent as a scalar, thereby neglecting dimensional information. For instance, these matrices might exhibit identical differences across various dimensions, an aspect that scalar representations fail to capture. This dimensional consistency is crucial for understanding the full impact of training data alterations on model outputs.

To further validate the effectiveness of our method described in Eq. 3.3, we analyze the difference between each method. In details, the KL divergence conducted in DAS can be seen

as:

$$D_{\mathrm{KL}}[p_\theta(\boldsymbol{x}^{\mathrm{gen}})||p_{\theta_{\setminus i}}(\boldsymbol{x}^{\mathrm{gen}})] = \int p_\theta(\boldsymbol{x}^{\mathrm{gen}}) \log p_\theta(\boldsymbol{x}^{\mathrm{gen}}) - p_\theta(\boldsymbol{x}^{\mathrm{gen}}) \log p_{\theta_{\setminus i}}(\boldsymbol{x}^{\mathrm{gen}}) dx. \quad (3.5)$$

This approach provides the most intuitive way of evaluating the difference between two distributions. In contrast, D-TRAK measures the difference between the model distributions and the data distribution , expressed as the KL divergence between $p_\theta(\boldsymbol{x}^{\mathrm{gen}})$, $p_{\theta_{\setminus i}}(\boldsymbol{x}^{\mathrm{gen}})]$ and $q(\boldsymbol{x}^{\mathrm{gen}})$, respectively:

$$\begin{aligned} & D_{\mathrm{KL}}[p_\theta(\boldsymbol{x}^{\mathrm{gen}}) \| q(\boldsymbol{x}^{\mathrm{gen}})] - D_{\mathrm{KL}}[p_{\theta_{\setminus i}}(\boldsymbol{x}^{\mathrm{gen}}) \| q(\boldsymbol{x}^{\mathrm{gen}})] \\ = & \int p_\theta(\boldsymbol{x}^{\mathrm{gen}}) \log p_\theta(\boldsymbol{x}^{\mathrm{gen}}) - p_{\theta_{\setminus i}}(\boldsymbol{x}^{\mathrm{gen}}) \log p_{\theta_{\setminus i}}(\boldsymbol{x}^{\mathrm{gen}}) dx - \\ & \int (p_\theta(\boldsymbol{x}^{\mathrm{gen}}) - p_{\theta_{\setminus i}}(\boldsymbol{x}^{\mathrm{gen}})) \log q(\boldsymbol{x}^{\mathrm{gen}}) dx. \end{aligned} \quad (3.6)$$

The computation of D-TRAK involves a term $\int (p_\theta(\boldsymbol{x}^{\mathrm{gen}}) - p_{\theta_{\setminus i}}(\boldsymbol{x}^{\mathrm{gen}})) \log q(\boldsymbol{x}^{\mathrm{gen}}) dx$, which represents the cross-entropy between the distribution shift and the data distribution. This term is generally nonzero unless $p_\theta(\boldsymbol{x}^{\mathrm{gen}})$ and $p_{\theta_{\setminus i}}(\boldsymbol{x}^{\mathrm{gen}})]$ are identical distributions. By including this term, D-TRAK involves the influence of the data distribution $q(\boldsymbol{x}^{\mathrm{gen}})$ in the attribution process, whereas DAS isolates the effect of the removed data on the model itself.

For Eq 3.4, which derived from Eq 3.2, it aims to empirically eliminate computations involving $q(\boldsymbol{x}^{\mathrm{gen}})$, instead focusing directly on the change in the model distribution. However, a comparison of Eq 3.4 with Eq. 3.3 reveals that, although Eq. 3.4 removes $q(\boldsymbol{x}^{\mathrm{gen}})$ from the computation, it does not consider the context in diffusion models. Specifically, it treats the output of noise predictor as a scalarrather than as a vector in latent space. This simplification introduces an error:

$$\tau_{\mathrm{DAS}} - \tau_{\mathrm{Square}} = 2\mathbb{E}_{\boldsymbol{\epsilon},t} < \boldsymbol{\epsilon}_{\theta_{\setminus i}}, \boldsymbol{\epsilon}_{\theta_{\setminus i}} - \boldsymbol{\epsilon}_\theta > . \quad (3.7)$$

τ_{Square} neglects the high-dimensional information embedded in the model output by omitting the inner product term, which involves cosine similarity information. By comparison, the difference between $\mathcal{L}_{\mathrm{Simple}}$ and $\mathcal{L}_{\mathrm{DAS}}$ is:

$$\tau_{\mathrm{DAS}} - \tau_{\mathrm{Simple}} = 2\mathbb{E}_{\boldsymbol{\epsilon},t} < \boldsymbol{\epsilon}, \boldsymbol{\epsilon}_\theta - \boldsymbol{\epsilon}_{\theta_{\setminus i}} > + 2\mathbb{E}_{\boldsymbol{\epsilon},t} < \boldsymbol{\epsilon}_{\theta_{\setminus i}}, \boldsymbol{\epsilon}_{\theta_{\setminus i}} - \boldsymbol{\epsilon}_\theta > . \quad (3.8)$$

In this case, τ_{Simple} further includes an additional cross term involving $\boldsymbol{\epsilon}$, which depends on its sampling. However, in practice, accurately approximating the expectation of $\boldsymbol{\epsilon}$ can be computationally expensive, leading to potential instability in calculating τ_{Simple} due to resource limitation.

3.2 Diffusion Attribution Score

Since we define the output change as KL-divergence, which differs from TRAK, f_{DAS} cannot be directly applied to TRAK and τ_{DAS} needs to be specifically derived on diffusion models. In this section, we explore methods to approximate Eq. 3.3 at timestep t without retraining the model. The derivation is divided into two main parts: First, we linearize the output function, allowing the difference in the output function to be expressed in terms of difference in model parameters. The second part is that approximating this relationship using Newton's method. By integrating these two components, we derive the complete formulation of DAS to attribute the output of diffusion model at timestep t.

Linearizing Output Function. Computing the output of the retrained model $\boldsymbol{\epsilon}_{\theta_{\setminus i}}(\boldsymbol{x}_t^{\text{gen}}, t)$ is computationally expensive. For computational efficiency, we propose linearizing the model output function around the optimal model parameters θ^* at convergence, simplifying the calculation as follows:

$$f_{\text{DAS}}(\boldsymbol{z}_t, \theta) \approx \boldsymbol{\epsilon}_{\theta^*}(\boldsymbol{x}_t, t) + \nabla_\theta \boldsymbol{\epsilon}_{\theta^*}(\boldsymbol{x}_t, t)^\top (\theta - \theta^*). \tag{3.9}$$

By substituting Eq. 3.9 into Eq. 3.3, we derive:

$$\tau_{\text{DAS}}(\boldsymbol{z}^{\text{gen}}, \mathbb{S})_t^{(i)} \approx \mathbb{E}_{\boldsymbol{\epsilon}}[||\nabla_\theta \boldsymbol{\epsilon}_{\theta^*}(\boldsymbol{x}_t^{\text{gen}}, t)^\top (\theta^* - \theta^*_{\setminus i})||^2]. \tag{3.10}$$

The subscript t indicates the attribution for the model output at timestep t. Consequently, the influence of removing a sample can be quantitatively evaluated through the changes in model parameters, which can be measured by the Newton's method, thereby reducing the computational overhead.

Estimating the model parameter. Consider using the leave-one-out method, the variation of the model parameters can be assessed by Newton's Method. In diffusion model, the parameter update process can be defined as:

$$\theta' \leftarrow \theta + \boldsymbol{H}_{\theta_t}^{-1}(\mathcal{L}_{\text{Simple}}(\theta))\nabla_\theta \mathcal{L}_{\text{Simple}}(\theta), \tag{3.11}$$

Here, $\boldsymbol{H}_{\theta_t}(\mathcal{L}_{\text{Simple}}(\theta))$ represents the Hessian matrix, and $\nabla_\theta \mathcal{L}_{\text{Simple}}$ is the gradient w.r.t the Simple Loss in the diffusion model. At convergence, the model reaches the global optimum parameter estimate θ^*, satisfying:

$$\boldsymbol{H}_{\theta_t}^{-1}(\mathcal{L}_{\text{Simple}}(\theta^*))\nabla_\theta \mathcal{L}_{\text{Simple}}(\theta^*) = 0. \tag{3.12}$$

Additionally, the Hessian matrix and gradient associated with the objective function at timestep t are defined as:

$$\mathbf{H}_{\theta^*} = \nabla_\theta \boldsymbol{\epsilon}_{\theta^*}(\mathbb{S}_t, t)^\top \nabla_\theta \boldsymbol{\epsilon}_{\theta^*}(\mathbb{S}_t, t),\ \nabla_\theta \mathcal{L}(\theta^*) = \nabla_\theta \boldsymbol{\epsilon}_{\theta^*}(\mathbb{S}_t, t)^\top \boldsymbol{R}_t, \tag{3.13}$$

where $\boldsymbol{\epsilon}_{\theta^*}(\mathbb{S}_t, t) := [\boldsymbol{\epsilon}_{\theta^*}(\boldsymbol{x}_t^{(1)}, t), ..., \boldsymbol{\epsilon}_{\theta^*}(\boldsymbol{x}_t^{(n)}, t)]$ denotes a stacked output matrix on $\mathbb{S}$ at timestep t and $\boldsymbol{R}_t := \text{diag}[\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t^{(i)}, t) - \boldsymbol{\epsilon}]$ is a diagonal matrix on $\mathbb{S}$ describing the residual among $\mathbb{S}$. Thus, the update defined in Eq. 3.11 around the optimum parameterθ^* is:

$$\theta' - \theta \leftarrow [\nabla_\theta \boldsymbol{\epsilon}_{\theta^*}(\mathbb{S}_t, t)^\top \nabla_\theta \boldsymbol{\epsilon}_{\theta^*}(\mathbb{S}_t, t)]^{-1} \nabla_\theta \boldsymbol{\epsilon}_{\theta^*}(\mathbb{S}_t, t)^\top \boldsymbol{R}_t. \tag{3.14}$$

Upon deleting a training sample $\boldsymbol{x}^{(i)}$ from $\mathbb{S}$, the counterfactual parameters $\theta^*_{\setminus i}$ can be estimated by applying a single step of Newton's method from the optimal parameter θ^* with the modified set $\mathbb{S}_{\setminus i}$, as follows:

$$\theta^* - \theta^*_{\setminus i} \leftarrow -[\nabla_\theta \boldsymbol{\epsilon}_{\theta^*}(\mathbb{S}_{\setminus i_t}, t)^\top \nabla_\theta \boldsymbol{\epsilon}_{\theta^*}(\mathbb{S}_{\setminus i_t}, t)]^{-1} \nabla_\theta \boldsymbol{\epsilon}_{\theta^*}(\mathbb{S}_{\setminus i_t}, t)^\top \boldsymbol{R}_{\setminus i_t}. \tag{3.15}$$

Let $\boldsymbol{g}_t(\boldsymbol{x}^{(i)}) = \nabla_\theta \boldsymbol{\epsilon}_{\theta^*}(\boldsymbol{x}_t^{(i)}, t)$ and $\boldsymbol{G}_t(\mathbb{S}) = \nabla_\theta \boldsymbol{\epsilon}_{\theta^*}(\mathbb{S}_t, t)$. The inverse term in Eq. 3.15 can be reformulated as:

$$\boldsymbol{G}_t(\mathbb{S}_{\setminus i})^\top \boldsymbol{G}_t(\mathbb{S}_{\setminus i}) = \boldsymbol{G}_t(\mathbb{S})^\top \boldsymbol{G}_t(\mathbb{S}) - \boldsymbol{g}_t(\boldsymbol{x}^{(i)})^\top \boldsymbol{g}_t(\boldsymbol{x}^{(i)}). \tag{3.16}$$

The Sherman-Morrison formula is defined as:

$$(\boldsymbol{A}+\boldsymbol{u}\boldsymbol{v}^\top)^{-1}=\boldsymbol{A}^{-1}-\frac{\boldsymbol{A}^{-1}\boldsymbol{u}\boldsymbol{v}^\top\boldsymbol{A}^{-1}}{1+\boldsymbol{v}^\top\boldsymbol{A}^{-1}\boldsymbol{u}}. \tag{3.17}$$

Let $\boldsymbol{H}=\boldsymbol{G}_t(\mathbb{S})^\top\boldsymbol{G}_t(\mathbb{S})$ and $\boldsymbol{u}=\boldsymbol{g}_t(\boldsymbol{x}^{(i)})$. Applying Eq. 3.17 in Eq. 3.16, we derive:

$$[\boldsymbol{G}_t(\mathbb{S}_{\backslash i})^\top\boldsymbol{G}_t(\mathbb{S}_{\backslash i})]^{-1}=[\boldsymbol{H}-\boldsymbol{u}\boldsymbol{u}^\top]^{-1}=\boldsymbol{H}^{-1}+\frac{\boldsymbol{H}^{-1}\boldsymbol{u}\boldsymbol{u}^\top\boldsymbol{H}^{-1}}{1-\boldsymbol{u}^\top\boldsymbol{H}^{-1}\boldsymbol{u}}. \tag{3.18}$$

Additionally, we have:

$$\boldsymbol{G}_t(\mathbb{S}_{\backslash i})^\top\boldsymbol{R}_{\backslash i_t}=\boldsymbol{G}_t(\mathbb{S})^\top\boldsymbol{R}_t-\boldsymbol{g}_t(\boldsymbol{x}^{(i)})^\top{\boldsymbol{r}^{(i)}}_t=-\boldsymbol{u}^\top\boldsymbol{r}_t^{(i)}, \tag{3.19}$$

where $\boldsymbol{r}_t^{(i)}$ is the i-th element of $\boldsymbol{R}_t$. Applying Eq. 3.18 and Eq. 3.19 to Eq. 3.15, we obtain:

$$\theta^*-\theta^*_{\backslash i}=[\boldsymbol{H}^{-1}+\frac{\boldsymbol{H}^{-1}\boldsymbol{u}\boldsymbol{u}^\top\boldsymbol{H}^{-1}}{1-\boldsymbol{u}^\top\boldsymbol{H}^{-1}\boldsymbol{u}}]\boldsymbol{u}^\top\boldsymbol{r}_t^{(i)}. \tag{3.20}$$

Let $\alpha=\boldsymbol{u}^\top\boldsymbol{H}^{-1}\boldsymbol{u}$. Eq. 3.20 simplifies to:

$$\begin{aligned}\theta^*-\theta^*_{\backslash i}&=\boldsymbol{H}^{-1}\boldsymbol{u}\cdot(1+\frac{\alpha}{1-\alpha})\boldsymbol{r}_t^{(i)}\\&=\boldsymbol{H}^{-1}\boldsymbol{u}\cdot\frac{1}{1-\alpha}\boldsymbol{r}_t^{(i)}\\&=\frac{[\boldsymbol{G}_t(\mathbb{S})^\top\boldsymbol{G}_t(\mathbb{S})]^{-1}\boldsymbol{g}_t(\boldsymbol{x}^{(i)})\boldsymbol{r}_t^{(i)}}{1-\boldsymbol{g}_t(\boldsymbol{x}^{(i)})^\top[(\boldsymbol{G}_t(\mathbb{S})^\top\boldsymbol{G}_t(\mathbb{S})]^{-1}\boldsymbol{g}_t(\boldsymbol{x}^{(i)})}.\end{aligned} \tag{3.21}$$

Applying the Sherman–Morrison formula to Eq. 3.16 simplifies Eq. 3.15 as follows:

$$\theta^*-\theta^*_{\backslash i}\leftarrow\frac{[\boldsymbol{G}_t(\mathbb{S})^\top\boldsymbol{G}_t(\mathbb{S})]^{-1}\boldsymbol{g}_t(\boldsymbol{x}^{(i)})\boldsymbol{r}_t^{(i)}}{1-\boldsymbol{g}_t(\boldsymbol{x}^{(i)})^\top[(\boldsymbol{G}_t(\mathbb{S})^\top\boldsymbol{G}_t(\mathbb{S})]^{-1}\boldsymbol{g}_t(\boldsymbol{x}^{(i)})}. \tag{3.22}$$

Diffusion Attribution Score. By substituting Eq.3.22 into Eq.3.10, we derive the formula for computing the DAS at timestep t:

$$\tau_{\text{DAS}}(\boldsymbol{z}^{\text{gen}},\mathbb{S})_t^{(i)}=\mathbb{E}_{\boldsymbol{\epsilon}}[||\frac{\boldsymbol{g}_t(\boldsymbol{x}^{\text{gen}})[\boldsymbol{G}_t(\mathbb{S})^\top\boldsymbol{G}_t(\mathbb{S})]^{-1}\boldsymbol{g}_t(\boldsymbol{x}^{(i)})\boldsymbol{r}_t^{(i)}}{1-\boldsymbol{g}_t(\boldsymbol{x}^{(i)})^\top[(\boldsymbol{G}_t(\mathbb{S})^\top\boldsymbol{G}_t(\mathbb{S})]^{-1}\boldsymbol{g}_t(\boldsymbol{x}^{(i)})}||^2]. \tag{3.23}$$

This equation estimates the impact of training samples at a specific timestep t. The overall influence of a training sample $\boldsymbol{z}^{(i)}$ on the target sample $\boldsymbol{z}^{\text{gen}}$ throughout the entire generation process can be computed as an expectation over timestep t. However, directly calculating

these expectations is extremely costly. In the next section, we discuss methods to expedite this computation.

3.3 Efficient DAS for large-scale diffusion models

Calculating Eq. 3.23 for large-scale diffusion models poses several challenges. The computation of the inverse term is extremely expensive due to the high dimensionality of the parameters. Additionally, gradients must be calculated for all training samples in $\mathbb{S}$, further increasing computational demands. In this section, we explore techniques to accelerate the calculation of Eq. 3.23. These methods can be broadly categorized into two approaches: The first focuses on reducing the gradient computation by minimizing the number of expectations and candidate training samples. The second aims to accelerate the computation of the inverse term by reducing the dimensionality of the gradients.

Reducing Calculation of Expectations. Computing t times the equation specified in Eq. 3.23 is highly resource-intensive due to the necessity of calculating inverse terms. To simplify, we use the average gradient $\overline{\boldsymbol{g}}(\boldsymbol{x})$ and average residual $\overline{\boldsymbol{r}}$ over entire generation, enabling a single computation of Eq.3.23 to assess overall influence. However, during averaging, these terms may exhibit varying magnitudes across different timesteps, potentially leading to the loss of significant information. To address this, we normalize the gradients and residuals over the entire generation before averaging:

$$\overline{\boldsymbol{g}}(\boldsymbol{x}^{(i)}) = \frac{1}{T}\sum_{t}\frac{\boldsymbol{g}_t(\boldsymbol{x}^{(i)})}{\sqrt{\sum_{j=1}^{T}[\boldsymbol{g}_j(\boldsymbol{x}^{(i)})]^2}},\ \overline{\boldsymbol{r}}^{(i)} = \frac{1}{T}\sum_{t}\frac{\boldsymbol{r}_t^{(i)}}{\sqrt{\sum_{j=1}^{T}[\boldsymbol{r}_j^{(i)}]^2}}. \tag{3.24}$$

Thus, to attribute the influence of a training sample $\boldsymbol{z}^{(i)}$ on a generated sample $\boldsymbol{z}^{\text{gen}}$ throughout the entire generation process, we redefine Eq. 3.23 as follows:

$$\tau_{\text{DAS}}(\boldsymbol{z}^{\text{gen}}, \mathbb{S})^{(i)} = ||\frac{\overline{\boldsymbol{g}}(\boldsymbol{x}^{\text{gen}})^\top[\overline{\boldsymbol{G}}(\mathbb{S})^\top\overline{\boldsymbol{G}}(\mathbb{S})]^{-1}\overline{\boldsymbol{g}}(\boldsymbol{x}^{(i)})\overline{\boldsymbol{r}}^{(i)}}{1-\overline{\boldsymbol{g}}(\boldsymbol{x}^{(i)})^\top[\overline{\boldsymbol{G}}(\mathbb{S})^\top\overline{\boldsymbol{G}}(\mathbb{S})]^{-1}\overline{\boldsymbol{g}}(\boldsymbol{x}^{(i)})}||^2. \tag{3.25}$$

Reducing Dimension of Gradients by Projection. The dimension of $\boldsymbol{g}_t(\boldsymbol{x}^{(i)})$ matches that of the amount of diffusion model's parameter, posing a challenge in calculating the inverse

term due to its substantial size. One effective method of reducing the dimensionalit is to apply the Johnson and Lindenstrauss Projection [31]. It involves multiplying the gradient vector $\boldsymbol{g}_t(\boldsymbol{x}^{(i)}) \in \mathbb{R}^p$ by a random matrix $\boldsymbol{P} \sim \mathcal{N}(0,1) \in \mathbb{R}^{p\times k}(k \ll p)$, which can preserve inner product with high probability while significantly reducing the dimension of the gradient. This projection method has been validated in previous studies [43, 22], demonstrating its efficacy in maintaining the integrity of the gradients while easing computational demands. We summarize our algorithms in Algorithm 1 with normalization and projection.

Algorithm 1 Diffusion Attribution Score

Input: Learning algorithm $\mathcal{A}$, Training dataset $\mathbb{S}$ of size n, Training data dimension p, Maximum timesteps for diffusion model T, Unet output in diffusion model $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}, t)$, Projection dimension k, A standard gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$, Normalization and average method N, A generated sample $\boldsymbol{z}^{\text{gen}}$
Output: Matrix of attribution scores $\boldsymbol{T} \in \boldsymbol{R}^n$
$\theta^* \leftarrow \mathcal{A}(\mathbb{S})$ {Train a diffusion model on $\mathbb{S}$}
$\boldsymbol{P} \sim \mathcal{N}(0,1)^{p\times k}$ {Sample projection matrix}
$\boldsymbol{R} \leftarrow 0_{n\times n}$
for $i = 1$ to n **do**
  for $t = 1$ to T **do**
    $\boldsymbol{\epsilon} \sim \mathcal{N}(0,1)^p$ {Sample a gaussian noise}
    $\boldsymbol{g}_t(\boldsymbol{x}^{(i)}) \leftarrow \boldsymbol{P}^\top \nabla_\theta \boldsymbol{\epsilon}_{\theta^*}(\boldsymbol{x}_t^{(i)}, t)$ {Compute gradient on training set and project}
    $\boldsymbol{r}_t^{(i)} \leftarrow \boldsymbol{\epsilon}_{\theta^*}(\boldsymbol{x}_t^{(i)}, t) - \boldsymbol{\epsilon}$ {Compute residual term}
  end for
  $\overline{\boldsymbol{g}}(\boldsymbol{x}^{(i)}) = N(\boldsymbol{g}_t(\boldsymbol{x}^{(i)}))$ {Normalize projected gradient term}
  $\overline{\boldsymbol{r}}^{(i)} = N(\boldsymbol{r}_t^{(i)})$ {Normalize residual term}
end for
$\overline{\boldsymbol{G}}(\mathbb{S}) \leftarrow [\overline{\boldsymbol{g}}(\boldsymbol{x}^{(i)}), ..., \overline{\boldsymbol{g}}(\boldsymbol{x}^{(n)})]^\top$
$\overline{\boldsymbol{R}} \leftarrow \text{diag}(\overline{\boldsymbol{r}}^{(1)}, ..., \overline{\boldsymbol{r}}^{(n)})$
for $t = 1$ to T **do**
  $\boldsymbol{\epsilon} \sim \mathcal{N}(0,1)^p$ {Sample a gaussian noise}
  $\boldsymbol{g}_t(\boldsymbol{x}^{\text{gen}}) \leftarrow \boldsymbol{P}^\top \nabla_\theta \boldsymbol{\epsilon}_{\theta^*}(\boldsymbol{x}_t^{\text{gen}}, t)$ {Compute gradient for generated sample and project}
end for
$\overline{\boldsymbol{g}}(x^{\text{gen}}) = N(\boldsymbol{g}_t(x^{\text{gen}}))$
$\boldsymbol{T} \leftarrow ||\frac{\overline{\boldsymbol{g}}(x^{\text{gen}})^\top (\overline{\boldsymbol{G}}(\mathbb{S})^\top \overline{\boldsymbol{G}}(\mathbb{S}))^{-1} \overline{\boldsymbol{G}}(\mathbb{S}) \overline{\boldsymbol{R}}}{1 - \overline{\boldsymbol{G}}(\mathbb{S})^\top (\overline{\boldsymbol{G}}(\mathbb{S})^\top \overline{\boldsymbol{G}}(\mathbb{S}))^{-1} \overline{\boldsymbol{G}}(\mathbb{S})}||^2$ {Compute attribution matrix}
return $(\boldsymbol{T})$

Reducing Dimension of Gradients by Model Compression. In addition to projection methods, other techniques can be employed to reduce the dimension of gradients in diffusion models. For instance, as noted by [44], the up-block of the U-Net architecture in diffusion

models plays a pivotal role in the generation process. Therefore, we can focus on the up-block gradients for dimension reduction purposes, optimizing computational efficiency. Furthermore, various strategies have been proposed to fine-tune large-scale diffusion models efficiently. One such approach is LoRA [45], which involves freezing the pre-trained model weights while utilizing trainable rank decomposition matrices. This significantly reduces the number of trainable parameters required for fine-tuning. Consequently, when attributing the influence of training samples in a fine-tuned dataset, we can compute the DAS with gradients on the trainable parameters.

Reducing the amount of timesteps. Computing Eq. 3.24 requires performing back propagation T times, making it highly resource-intensive. Sampling fewer timesteps can also approximate the expectation and estimate gradient behavior while significantly lowering computational overhead.

Reducing Candidate Training Sample. The necessity to traverse the entire training set when computing the DAS poses a significant challenge. To alleviate this, a practical approach involves conducting a preliminary screening to identify the most influential training samples. Techniques such as CLIP [46] or cosine similarity can be effectively employed to locate samples that are similar to the target. By using these methods, we can form a preliminary candidate set and concentrate DAS computations on this subset, rather than on the entire training dataset.

CHAPTER 4

Results

4.1 Datasets and Models

In this section, we present a comparative analysis of our proposed method, Diffusion Attribution Score (DAS), against existing data attribution approaches across various experimental settings. Our findings highlight that DAS significantly outperforms other methods in attribution performance, validating its effectiveness in accurately identifying influential training samples. Below, we provide a comprehensive overview of the datasets and diffusion models used in our experiments.

CIFAR10(32×32). The CIFAR-10 dataset [47] comprises 50,000 training images spanning 10 classes. For efficiency in ablation studies, we introduce CIFAR-2, a subset containing 5,000 training samples and 1,000 validation samples, specifically drawn from the "automobile" and "horse" categories. For evaluation using the Linear Datamodeling Score (LDS), we also utilize a subset of 1,000 images randomly selected from CIFAR-10's test set. In our experiments, we employ Denoising Diffusion Probabilistic Models (DDPMs)[2] configured with approximately 35.7 million parameters. The forward diffusion process uses a linear variance schedule with $\beta_1 = 10^{-4}$ and $\beta_T = 0.02$, spanning a maximum of 1,000 timesteps. A 50-step DDIM solver[48] is used for inference. Key hyperparameters include a dropout rate of 0.1 and the AdamW optimizer [49] with a weight decay of 10^{-6}. The model is trained for 200 epochs with a batch size of 128, leveraging a cosine annealing learning rate schedule with a 10% warm-up period, starting from an initial rate of 10^{-4}. Data augmentation, including random horizontal flips, is applied to enhance robustness.

CelebA (64×64). The CelebA dataset [50] is utilized with 5,000 training samples and 1,000 validation samples, extracted from the original training and test sets. Images are preprocessed following the steps in [3], where they are center-cropped to 140×140 pixels and resized to 64×64 pixels.

For this dataset, we adopt a similar DDPM architecture as in CIFAR-10 experiments, but the U-Net architecture is expanded to 118.8 million parameters to accommodate the increased complexity of CelebA. The variance schedule, optimizer settings, and training protocol mirror those used for CIFAR-10. During inference, the 50-step DDIM solver is used for image generation.

ArtBench (256×256). ArtBench [51] is a dataset of 60,000 images spanning 10 artistic styles. Two subsets are introduced for evaluation: ArtBench-2, with 5,000 training and 1,000 validation samples from the "post-impressionism" and "ukiyo-e" styles, and ArtBench-5, with 12,500 training and 1,000 validation samples across five styles: "post-impressionism," "ukiyo-e," "romanticism," "renaissance," and "baroque." We fine-tune a Stable Diffusion model [8] on these datasets using Low-Rank Adaptation (LoRA)[45] with a rank of 128, resulting in 25.5 million parameters. The model is adapted from a pre-trained Stable Diffusion checkpoint at a resolution of 512×512, resized to 256×256 to align with the dataset. Training is conditioned on textual prompts specific to each style (e.g., "a romanticism painting"). The dropout rate is set to 0.1, and the AdamW optimizer is used with a weight decay of 10^{-6}. Data augmentation includes random horizontal flips, and the training spans 100 epochs with a batch size of 64. The learning rate follows a cosine annealing schedule, starting at 3×10^{-4} with a 10% warm-up period. During inference, images are generated using a 50-step DDIM solver with a classifier-free guidance scale of 7.5 [52].

4.2 Evaluation Method for Data attribution

A variety of methods are available to evaluate data attribution techniques, including the leave-one-out influence method [9, 29] and Shapley values [53]. In this paper, we adopt the Linear Datamodeling Score (LDS) [11] as the primary evaluation metric due to its ability to

quantify the alignment between attribution-based predictions and actual model outputs in a principled and robust manner. LDS evaluates a data attribution method τ by sampling a subset $\mathbb{S}' \subset \mathbb{S}$ from the training dataset $\mathbb{S}$ and retraining a new model θ' on $\mathbb{S}'$. The attribution-based predicted output for a test sample $\boldsymbol{z}^{\text{test}}$ is computed as:

$$g_\tau(\boldsymbol{z}^{\text{test}}, \mathbb{S}', \mathbb{S}) := \sum_{\boldsymbol{z}^{(i)} \in \mathbb{S}'} \tau(\boldsymbol{z}^{\text{test}}, \mathbb{S})^{(i)}. \tag{4.1}$$

The core assumption of LDS is that the predicted output $g_\tau(\boldsymbol{z}, \mathbb{S}', \mathbb{S})$ should closely approximate the actual model output $f(\boldsymbol{z}^{\text{test}}, \theta')$. To evaluate this correspondence, LDS calculates the Spearman correlation (ρ) between attribution-based predictions and actual model outputs across M sampled subsets:

$$\text{LDS}(\tau, \boldsymbol{z}^{\text{test}}) := \rho\left(f(\boldsymbol{z}^{\text{test}}, \theta_m) : m \in [M], g_\tau(\boldsymbol{z}^{\text{test}}, \mathbb{S}^m, \mathbb{S}) : m \in [M]\right), \tag{4.2}$$

where $\theta_m$is the model retrained on the m-th subset $\mathbb{S}_m$. For our LDS evaluation, we construct 64 subsets $\mathbb{S}_m$ from the training dataset $\mathbb{S}$, each containing 50% of the training samples. For CIFAR-2, ArtBench-2, and ArtBench-5, three models are trained per subset using different random seeds to ensure robustness, while for CIFAR-10 and CelebA, a single model is trained per subset. The validation set, comprising 1,000 samples from both the original test set and a generated dataset, serves as the test data $\boldsymbol{z}^{\text{test}}$ for LDS calculations. To maintain consistency with D-TRAK [22], we use the Simple Loss $\mathcal{L}_{\text{Simple}}(\boldsymbol{z}, \theta)$ defined in Eq. 2.3 as the output function for all evaluations. To approximate the expectations over timesteps ($\mathbb{E}_t$) and noise ($\mathbb{E}_{\boldsymbol{\epsilon}}$), we utilize 1,000 evenly spaced timesteps from $[1, T]$ and sample three instances of standard Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$ at each timestep. The final LDS values are averaged across all selected validation and generation samples to compute the overall performance. This evaluation framework ensures a comprehensive and robust analysis of data attribution methods, measuring their ability to predict model behavior accurately while aligning with actual retrained model outputs.

TABLE 4.1. We compare D-TRAK and our methods DAS with the normalization and without normalization on CIFAR2. Besides, we also select 10, 100 and 1000 timesteps evenly spaced within the interval $[1, T]$ and calculate the average of LDS(%) among the timesteps.

Method	Normalization	Validation			Generation		
		10	100	1000	10	100	1000
D-TRAK	No Normalization	24.78	30.81	32.37	16.20	22.62	23.94
	Normalization	26.11	31.50	32.51	17.09	22.92	24.10
DAS	No Normalization	33.04	42.02	43.13	20.01	29.58	30.58
	Normalization	33.77	42.26	43.28	21.24	29.60	30.87

TABLE 4.2. We compare our methods with TRAK and D-TRAK by LDS method on CIFAR-2 among different selected timesteps. The projected dimension $k = 4096$.

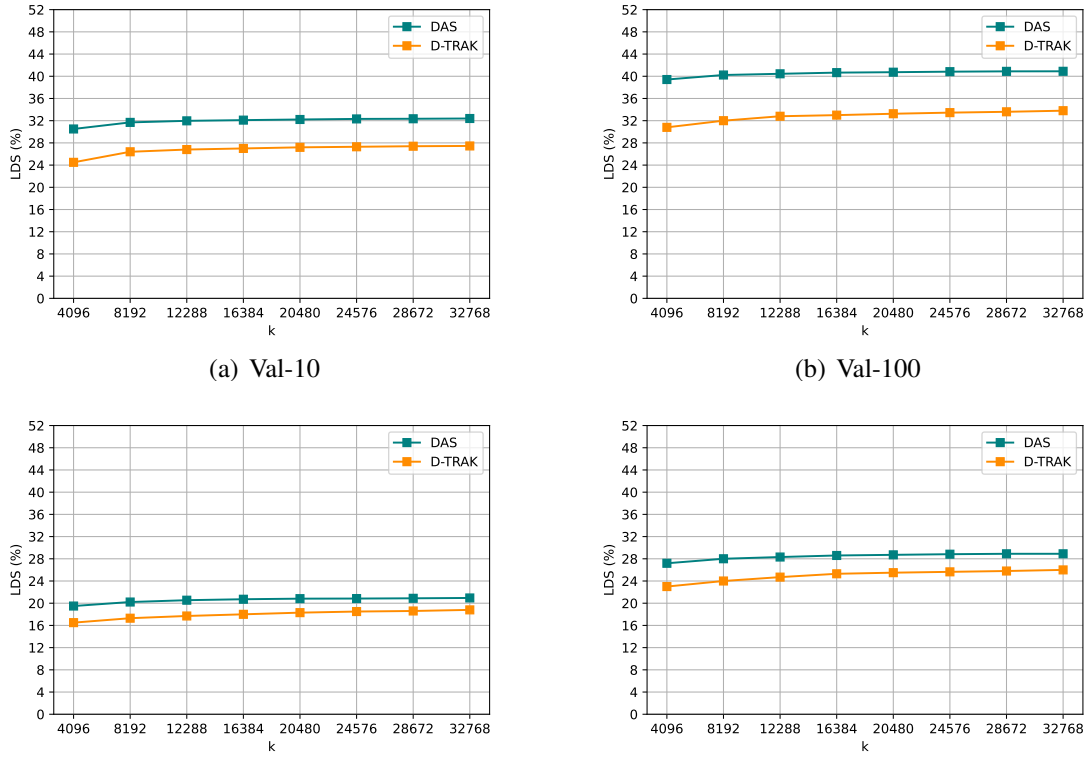
Method	Validation			Generation		
	10	100	1000	10	100	1000
TRAK	10.66	19.50	22.42	5.14	12.05	15.46
D-TRAK	24.91	30.91	32.39	16.76	22.62	23.94
DAS	**33.04**	**42.02**	**43.13**	**20.01**	**29.58**	**30.58**

4.3 Evaluation for Speed Up Techniques

The diffusion models used in our experiments are significantly complex, with parameter counts of 35.7M, 118.8M, and 25.5M respectively. These large dimensions pose considerable challenges in calculating the attribution score efficiently. To address this, we evaluate speed-up techniques as discussed in Section 3.3 on CIFAR-2.

Normalization. We evaluate the normalization of gradients and residuals, as proposed in Eq. 3.24, to stabilize gradient variability across timesteps and enhance computational accuracy. By normalizing across generation before averaging, the performance for both DAS and D-TRAK improve (Table 4.1).

Number of timesteps. Computing DAS requires balancing effectiveness and computational efficiency, as more timesteps improve performance through averaging but increase back-propagation costs. Experiment shows that while increasing timesteps enhances LDS results



(a) Val-10 (b) Val-100

(c) Gen-10 (d) Val-100

FIGURE 4.1. The LDS(%) on CIFAR-2 under different projection dimension k. We consider 10 and 100 timesteps selected to be evenly spaced within the interval $[1, T]$, which are used to approximate the expectation $\mathbb{E}_t$. For each sampled timestep, we sample one standard Gaussian noise $\epsilon \sim \mathcal{N}(\epsilon|0, I)$ to approximate the expectation $\mathbb{E}_\epsilon$.

(Table 4.2), using 100 or 10 timesteps achieves comparable performance to 1000 timesteps with much lower computational demands. Thus, subsequent experiments will default to 10 and 100 timesteps for optimal efficiency.

Projection. We apply the projection technique to reduce gradient dimensions and analyzed the impact of projection dimension k on LDS performance. As [31] disucssed, higher projection dimensions better preserve inner products but increase computational costs. Figure 4.1 shows that LDS scores for both D-TRAK and DAS improve with increasing k before plateauing. Based on these results, we set $k = 32768$ as the default for experiments.

TABLE 4.3. We compute DAS only with the Up-Block gradients in U-Net and evaluate by LDS method on CIFAR-2 among different selected timesteps. The projected dimension $k = 32768$.

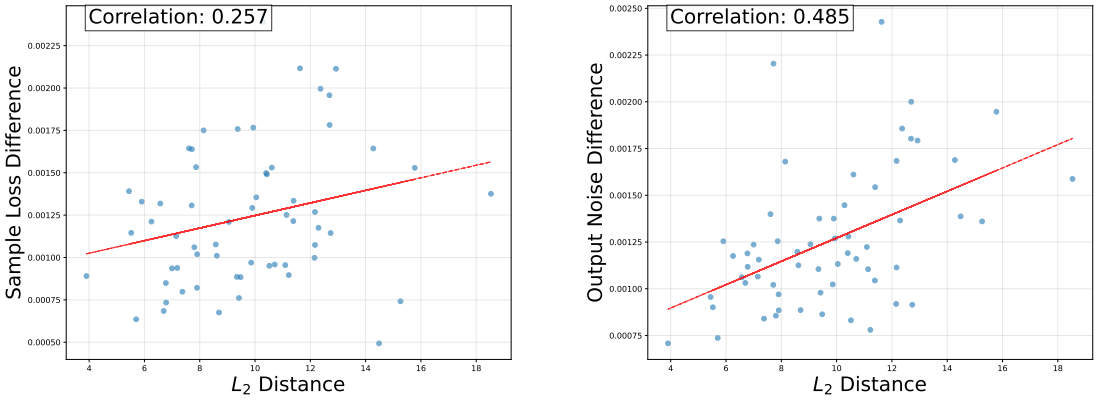
Method	Validation		Generation	
	10	100	10	100
D-TRAK	24.91	30.91	16.76	22.62
DAS(Up-Block)	32.60	37.90	18.47	27.54
DAS(U-Net)	**33.77**	**42.26**	**21.24**	**29.60**

TABLE 4.4. We compute DAS only with the Up-Block gradients in U-Net and evaluate by LDS method on CIFAR-2 among different selected timesteps. The projected dimension $k = 32768$.

Method	Validation		Generation	
	10	100	10	100
D-TRAK	24.91	30.91	16.76	22.62
DAS(Candidate Set)	31.53	37.75	17.73	23.31
DAS(Entire Training Set)	**33.77**	**42.26**	**21.24**	**29.60**

Compress Model Parameters. We also explore techniques to reduce the gradient dimension at the model level. We conduct experiments focusing on using the up-block of the U-Net to compute gradients. The results in Table 4.3 indicate that using only the up-block can achieve competitive performance compared to the full model . Moreover, experiments on ArtBench, which fine-tune a Stable Diffusion model with LoRA, further demonstrated the effectiveness of this approach.

Candidate Training Sample. Another technique to speed up the process involves reducing the number of training samples considered. We using CLIP to identify the top 1,000 training samples most similar to the target samples to form a candidate dataset We then computed the attribution scores for this candidate set, assigning a score of 0 to all other samples, and calculated the LDS. The results, detailed in Table 4.4, validate the efficacy of this method.



(a) Origin (b) Counter Factual

FIGURE 4.2. The result of Toy Experiment in Section 4.4. Scatter plot showing the relationship between L^2 distance and two metrics: loss difference and noise predictor output difference over entire generation. The noise predictor output difference exhibits a stronger correlation with L^2 distance, indicating its effectiveness in capturing image variation.

(a) Origin (b) Counter Factual (c) Origin (d) Counter Factual

FIGURE 4.3. Figures 4.3(a) and 4.3(b) represent a pair of generated images, while 4.3(c) and 4.3(d) form another pair. Despite the loss difference between 4.3(a) and 4.3(b) being only 0.0007, the L^2 distance is 15.261. Similarly, the loss difference between 4.3(c) and 4.3(d) is also 0.0007, yet the L^2distance is 14.485, showing that the loss value fail to trace the change on generated images.

4.4 Evaluating Output Function Effectiveness

In this paper, we define the output function as $f_{\text{DAS}} = \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t^{\text{gen}}, t)$ to evaluate differences between predicted distributions after data intervention. We argue that using the Simple

Loss for this purpose introduces error, as it indirectly reflects the data distribution's impact during attribution rather than directly capturing changes in generated images. To validate this theoretical claim, we conduct a toy experiment to compare the alignment of changes in the Simple Loss and the noise predictor's output with variations in generated images.

In the experiment, we train an unconditional DDPM on CIFAR-2 to generate 60 images. For each image, we retrain the model after randomly removing 1,000 training samples, resulting in 60 pairs of generated images produced by the original and retrained models under the same random seed. The L^2 distance is computed between each pair to directly measure the differences in the generated outputs. The original images are first noised to timestep T, and both models are used to denoise the latent variables at T. Throughout the denoising process, we calculate the average differences in the Simple Loss and the noise predictor's output.

For the 60 pairs of generated samples, we compute the Pearson correlation between the L^2 distance and each of these differences. The results show that the correlation between the L^2 distance and the Simple Loss difference is only 0.257, while the correlation with the average noise predictor output difference reaches 0.485. This indicates that the noise predictor's output aligns more closely with the L^2 distance, effectively capturing changes in generated images. The results of this experiment are visualized in Figure 4.2.

To further illustrate, Figure 4.3 presents specific examples. For image pairs (4.3(a), 4.3(b)) and (4.3(c), 4.3(d)), the loss differences between models are only 0.0007, the smallest among all pairs. However, the L^2 distances between images in these pairs are 15.261 and 14.485, respectively, ranking among the largest in the dataset. This demonstrates that Simple Loss differences fail to reflect significant changes in the images. In contrast, the noise predictor's output differences for these pairs align closely with the overall trend in Figure 4.2(b), showing a strong correlation with the L^2 distances.

These findings are consistent with our theoretical claim. Consider an extreme scenario where a model trained on a dataset of cats and dogs generates a cat image. If all cat samples are removed and the model is retrained, it would generate a dog image under the same random seed. Despite this significant change, the loss values for both images might remain the same,

as the Simple Loss reflects model optimization rather than the actual generated image. In contrast, analyzing the differences in the noise predictor's outputs at each timestep allows us to effectively trace changes in the diffusion model's outputs, providing a more reliable measure of variation in generated images.

4.5 Baseline Methods

In this paper, our focus is primarily on post-hoc data attribution, which entails applying attribution methods after the completion of model training. These methods are particularly advantageous as they do not impose additional constraints during the model training phase, making them well-suited for practical applications [54].

Following the work of [55], we evaluate various attribution baselines that are compatible with our experimental framework. We exclude certain methods that are not feasible for our settings, such as the Leave-One-Out approach [56] and the Shapley Value method [17, 18]. These methods, although foundational, do not align well with the requirements of DDPMs due to their intensive computational demands and model-specific limitations. Additionally, we do not consider techniques like Representer Point [10], which are tailored for specific tasks and models, and thus are incompatible with DDPMs. Moreover, we disregard HYDRA [57], which, although related to TracInCP [20], compromises precision for incremental speed improvements as critiqued by [55].

Two works focus on diffusion model that also fall outside our framework. [32] proposes a method for training data attribution on diffusion models using machine unlearning; however, their approach necessitates a specific machine unlearning training process, making it non-post-hoc and thus unsuitable for standard settings. Similarly, [33] acknowledge the current challenges in conducting post-hoc training influence analysis with existing methods. They suggest an alternative termed "customization," which involves adapting or tuning a pretrained text-to-image model through a specially designed training procedure.

Building upon recent advancements, [21] introduced an innovative estimator that leverages a kernel matrix analogous to the Fisher Information Matrix (FIM), aiming to linearize the model's behavior. This approach integrates classical random projection techniques to expedite the computation of Hessian-based influence functions [9], which are typically computationally intensive. [22] adapted TRAK to diffusion models, empirically designing the model output function. Intriguingly, they reported that the theoretically designed model output function in TRAK performs poorly in unsupervised settings within diffusion models. However, they did not provide a theoretical explanation for these empirical findings, leaving a gap in understanding the underlying mechanics.

Our study concentrates on retraining-free methods, which we categorize into three distinct types: similarity-based, gradient-based (without kernel), and gradient-based (with kernel) methods. For similarity-based approaches, we consider Raw pixel similarity and CLIP similarity [46]. The gradient-based methods without a kernel include techniques such as Gradient [58], TracInCP [20] and GAS [59]. In the domain of gradient-based methods with a kernel, we explore several methods including D-TRAK [22], TRAK [21], Relative Influence [60], Renormalized Influence [59], and Journey TRAK [35].

We next provide definition and implementation details of the baselines used in Section 4.6.

TracInCP. We implement the TracInCP estimator, as outlined by [20], which quantifies the influence of training samples using the following formula:

$$\tau(\boldsymbol{z}, \mathbb{S})^i = \frac{1}{C}\Sigma_{c=1}^{C}(\boldsymbol{P}_c^\top \nabla_\theta \mathcal{L}_{\text{Simple}}(\boldsymbol{x}, \theta^c))^\top \cdot (\boldsymbol{P}_c^\top \nabla_\theta \mathcal{L}_{\text{Simple}}(\boldsymbol{x}^i, \theta^c)),$$

where C represents the number of model checkpoints selected evenly from the training trajectory, and θ^c denotes the model parameters at each checkpoint. For our analysis, we select four specific checkpoints along the training trajectory to ensure a comprehensive evaluation of the influence over different phases of learning. For example, in the CIFAR-2 experiment, the chosen checkpoints occur at epochs 50, 100, 150, and 200, capturing snapshots of the model's development and adaptation.

GAS. The GAS method is essentially a "renormalized" version of TracInCP that employs cosine similarity for estimating influence, rather than relying on raw dot products. This method was introduced by [59] and aims to refine the estimation of influence by normalizing the gradients. This approach allows for a more nuanced comparison between gradients, considering not only their directions but also normalizing their magnitudes to focus solely on the directionality of influence.

TRAK. The retraining-free version of TRAK [21] utilizes a model's trained state to estimate the influence of training samples without the need for retraining the model at each evaluation step. This version is implemented using the following equations:

$$\boldsymbol{\Phi}_{\text{TRAK}} = \left[\boldsymbol{\Phi}(\boldsymbol{x}^1), \cdots, \boldsymbol{\Phi}(\boldsymbol{x}^N)\right]^\top, \text{where } \boldsymbol{\Phi}(\boldsymbol{x}) = \boldsymbol{P}^\top \nabla_\theta \mathcal{L}_{\text{Simple}}(\boldsymbol{x}, \theta),$$
$$\tau(\boldsymbol{z}, \mathbb{S})^i = (\boldsymbol{P}^\top \nabla_\theta \mathcal{L}_{\text{Simple}}(\boldsymbol{x}, \theta))^\top \cdot \left({\boldsymbol{\Phi}_{\text{TRAK}}}^\top \boldsymbol{\Phi}_{\text{TRAK}} + \lambda \boldsymbol{I}\right)^{-1} \cdot \boldsymbol{P}^\top \nabla_\theta \mathcal{L}_{\text{Simple}}(\boldsymbol{x}^i, \theta),$$

where $\lambda \boldsymbol{I}$ is included for numerical stability and regularization.

D-TRAK. Simliar to TRAK, we adapt the D-TRAK [22] as detailed in Eq 2.6. We implent the model output function $f(\boldsymbol{z}, \theta)$ as $\mathcal{L}_{\text{Square}}$. The D-TRAK is implemented using the following equations:

$$\boldsymbol{\Phi}_{\text{D-TRAK}} = \left[\boldsymbol{\Phi}(\boldsymbol{x}^1), \cdots, \boldsymbol{\Phi}(\boldsymbol{x}^N)\right]^\top, \text{where } \boldsymbol{\Phi}(\boldsymbol{x}) = \boldsymbol{P}^\top \nabla_\theta \mathcal{L}_{\text{Simple}}(\boldsymbol{x}, \theta),$$
$$\tau(\boldsymbol{z}, \mathbb{S})^i = (\boldsymbol{P}^\top \nabla_\theta \mathcal{L}_{\text{Simple}}(\boldsymbol{x}, \theta))^\top \cdot \left({\boldsymbol{\Phi}_{\text{TRAK}}}^\top \boldsymbol{\Phi}_{\text{TRAK}} + \lambda \boldsymbol{I}\right)^{-1} \cdot \boldsymbol{P}^\top \nabla_\theta \mathcal{L}_{\text{Simple}}(\boldsymbol{x}^i, \theta),$$

where λI is also included for numerical stability and regularization as TRAK. Additionally, the output function $f(\boldsymbol{z}, \theta)$ could be replaced to other functions.

Relative Influence. [60] introduce the θ-relative influence functions estimator, which normalizes the influence functions estimator from [9] by the magnitude of the Hessian-vector product (HVP). This normalization enhances the interpretability of influence scores by adjusting for the impact magnitude. We have adapted this method to our experimental framework by incorporating scalability optimizations from TRAK. The adapted equation for the Relative Influence is formulated as follows:

$$\tau(\boldsymbol{z}, \mathbb{S})^{(i)} = \frac{(\boldsymbol{P}^\top \nabla_\theta \mathcal{L}_{\text{Simple}}(\boldsymbol{x}, \theta))^\top \cdot \left({\boldsymbol{\Phi}_{\text{ TRAK}}}^\top \boldsymbol{\Phi}_{\text{ TRAK}} + \lambda \boldsymbol{I}\right)^{-1} \cdot \boldsymbol{P}^\top \nabla_\theta \mathcal{L}_{\text{Simple}}(\boldsymbol{x}^{(i)}, \theta^*)}{|| \left({\boldsymbol{\Phi}_{\text{TRAK}}}^\top \boldsymbol{\Phi}_{\text{ TRAK}} + \lambda I\right)^{-1} \cdot \boldsymbol{P}^\top \nabla_\theta \mathcal{L}_{\text{Simple}}(\boldsymbol{x}^{(i)}, \theta^*)||}.$$

Renormalized Influence. [59] propose a method to renormalize influence by considering the magnitude of the training sample's gradients. This approach emphasizes the relative strength of each sample's impact on the model, making the influence scores more interpretable and contextually relevant. We have adapted this method to our settings by incorporating TRAK's scalability optimizations, which are articulated as:

$$\tau(\boldsymbol{z}, \mathbb{S})^{(i)} = \frac{(\boldsymbol{P}^\top \nabla_\theta \mathcal{L}_{\text{Simple}}(\boldsymbol{x}, \theta))^\top \cdot \left(\boldsymbol{\Phi}_{\text{TRAK}}{}^\top \boldsymbol{\Phi}_{\text{TRAK}} + \lambda \boldsymbol{I}\right)^{-1} \cdot \boldsymbol{P}^\top \nabla_\theta \mathcal{L}_{\text{Simple}}(\boldsymbol{x}^{(i)}, \theta)}{||\boldsymbol{P}^\top \nabla_\theta \mathcal{L}_{\text{Simple}}(\boldsymbol{x}^{(i)}, \theta)||}.$$

Journey TRAK. Journey TRAK [35] focuses on attributing influence to noisy images $\boldsymbol{x}_t$ at a specific timestep t throughout the generative process. In contrast, our approach aims to attribute the final generated image $\boldsymbol{x}^{\text{gen}}$, necessitating an adaptation of their method to our context. We average the attributions across the generation timesteps, detailed in the following equation:

$$\tau(\boldsymbol{z}, \mathbb{S})^{(i)} = \frac{1}{T'} \Sigma_{t=1}^{T'} (\boldsymbol{P}^\top \nabla_\theta \mathcal{L}^t_{\text{Simple}}(\boldsymbol{x}_t, \theta))^\top \cdot \left(\boldsymbol{\Phi}_{\text{TRAK}}{}^\top \boldsymbol{\Phi}_{\text{TRAK}} + \lambda \boldsymbol{I}\right)^{-1} \cdot \boldsymbol{P}^\top \nabla_\theta \mathcal{L}_{\text{Simple}}(\boldsymbol{x}^{(i)}, \theta),$$

where T' represents the number of inference steps, set at 50, and $\boldsymbol{x}_t$ denotes the noisy image generated along the sampling trajectory.

Raw pixel. This method employs a naive similarity-based approach for data attribution by using the raw image data itself as the representation. Specifically, for experiments on ArtBench, which utilizes latent diffusion models [8], we represent the images through the VAE encodings [61] of the raw image. The attribution score is calculated by computing either the dot product or cosine similarity between the sample of interest and each training sample, facilitating a straightforward assessment of similarity based on pixel values.

CLIP Similarity. This method represents another similarity-based approach to data attribution. Each sample is encoded into an embedding using the CLIP model [46], which captures semantic and contextual nuances of the visual content. The attribution score is then determined by computing either the dot product or cosine similarity between the CLIP embedding of the target sample and those of the training samples. This method leverages

TABLE 4.5. LDS (%) on CIFAR-2/CIFAR-10 with timesteps (10 or 100).

Method	CIFAR2				CIFAR10			
	Validation		Generation		Validation		Generation	
	10	100	10	100	10	100	10	100
Raw pixel (dot prod.)	7.77±0.57		4.89±0.58		2.50±0.42		2.25±0.39	
Raw pixel (cosine)	7.87±0.57		5.44±0.57		2.71±0.41		2.61±0.38	
CLIP similarity (dot prod.)	6.51±1.06		3.00±0.95		2.39±0.41		1.11±0.47	
CLIP similarity (cosine)	8.54±1.01		4.01±0.85		3.39±0.38		1.69±0.49	
Gradient (dot prod.)	5.14±0.60	5.07±0.55	2.80±0.55	4.03±0.51	0.79±0.43	1.40±0.42	0.74±0.45	1.85±0.54
Gradient (cosine)	5.08±0.59	4.89±0.50	2.78±0.54	3.92±0.49	0.66±0.43	1.24±0.41	0.58±0.42	1.82±0.51
TracInCP	6.26±0.84	5.47±0.87	3.76±0.61	3.70±0.66	0.98±0.44	1.26±0.38	0.96±0.40	1.39±0.54
GAS	5.78±0.82	5.15±0.87	3.34±0.56	3.30±0.68	0.89±0.48	1.25±0.38	0.90±0.41	1.61±0.54
Journey TRAK	/	/	7.73±0.65	12.21±0.46	/	/	3.71±0.37	7.26±0.43
Relative IF	11.20±0.51	23.43±0.46	5.86±0.48	15.91±0.39	2.76±0.45	13.56±0.39	2.42±0.36	10.65±0.42
Renorm. IF	10.89±0.46	21.46±0.42	5.69±0.45	14.65±0.37	2.73±0.46	12.58±0.40	2.10±0.34	9.34±0.43
TRAK	11.42±0.49	23.59±0.46	5.78±0.48	15.87±0.39	2.93±0.46	13.62±0.38	2.20±0.38	10.33±0.42
D-TRAK	26.79±0.33	33.74±0.37	18.82±0.43	25.67±0.40	14.69±0.46	20.56±0.42	11.05±0.43	16.11±0.36
DAS	**33.90±0.69**	**43.08±0.37**	**20.88±0.27**	**30.68±0.76**	**24.74±0.41**	**33.23±0.35**	**15.24±0.51**	**23.69±0.47**

the rich representational power of CLIP embeddings to ascertain the contribution of training samples to the generation or classification of new samples.

Gradient. This method employs a gradient-based approach to estimate the influence of training samples, as described by [58]. The attribution score is calculated by taking the dot product or cosine similarity between the gradients of the sample of interest and those of each training sample. This technique quantifies how much the gradient (indicative of the training sample's influence on the loss) of a particular training sample aligns with the gradient of the sample of interest, providing insights into which training samples most significantly affect the model's output.

4.6 Evaluating LDS for Various Attribution Methods

In this section, we evaluate the performance of the Diffusion Attribution Score (DAS) against existing attribution baselines applicable to our experimental settings, as outlined by [22], with detailed explanations provided in Section 4.5. Our primary focus is on comparing with post-hoc data attribution methods. To ensure fair comparisons, we limit the use of acceleration techniques for DAS to projection only, excluding others like normalization. The results on CIFAR, ArtBench and CelebA, presented in Tables 4.5, 4.6, demonstrate that DAS

TABLE 4.6. LDS (%) on ArtBench-2/ArtBench-5 with timesteps (10 or 100)

Method	ArtBench2				ArtBench5			
	Validation		Generation		Validation		Generation	
	10	100	10	100	10	100	10	100
Raw pixel (dot prod.)	2.44±0.56		2.60±0.84		1.84±0.42		2.77±0.80	
Raw pixel (cosine)	2.58±0.56		2.71±0.86		1.97±0.41		3.22±0.78	
CLIP similarity (dot prod.)	7.18±0.70		5.33±1.45		5.29±0.45		4.47±1.09	
CLIP similarity (cosine)	8.62±0.70		8.66±1.31		6.57±0.44		6.63±1.14	
Gradient (dot prod.)	7.68±0.43	16.00±0.51	4.07±1.07	10.23±1.08	4.77±0.36	10.02±0.45	3.89±0.88	8.17±1.02
Gradient (cosine)	7.72±0.42	16.04±0.49	4.50±0.97	10.71±1.07	4.96±0.35	9.85±0.44	4.14±0.86	8.18±1.01
TracInCP	9.69±0.49	17.83±0.58	6.36±0.93	13.85±1.01	5.33±0.37	10.87±0.47	4.34±0.84	9.02±1.04
GAS	9.65±0.46	18.04±0.62	6.74±0.82	14.27±0.97	5.52±0.38	10.71±0.48	4.48±0.83	9.13±1.01
Journey TRAK	/	/	5.96±0.97	11.41±1.02	/	/	7.59±0.78	13.31±0.68
Relative IF	12.22±0.43	27.25±0.34	7.62±0.57	19.78±0.69	9.77±0.34	20.97±0.41	8.89±0.59	19.56±0.62
Renorm. IF	11.90±0.43	26.49±0.34	7.83±0.64	19.86±0.71	9.57±0.32	20.72±0.40	8.97±0.58	19.38±0.66
TRAK	12.26±0.42	27.28±0.34	7.78±0.59	20.02±0.69	9.79±0.33	21.03±0.42	8.79±0.59	19.54±0.61
D-TRAK	27.61±0.49	32.38±0.41	24.16±0.67	26.53±0.64	22.84±0.37	27.46±0.37	21.56±0.71	23.85±0.71
DAS	**37.96±0.64**	**40.77±0.47**	**30.81±0.31**	**32.31±0.42**	**35.33±0.49**	**37.67±0.68**	**31.74±0.75**	**32.77±0.53**

TABLE 4.7. LDS (%) on CelebA with timesteps (10 or 100)

Results on CelebA				
Method	Validation		Generation	
	10	100	10	100
Raw pixel (dot prod.)	5.58±0.73		-4.94±1.58	
Raw pixel (cosine)	6.16±0.75		-4.38±1.63	
CLIP similarity (dot prod.)	8.87±1.14		2.51±1.13	
CLIP similarity (cosine)	10.92±0.87		3.03±1.13	
Gradient (dot prod.)	3.82±0.50	4.89±0.65	3.83±1.06	4.53±0.84
Gradient (cosine)	3.65±0.52	4.79±0.68	3.86±0.96	4.40±0.86
TracInCP	5.14±0.75	4.89±0.86	5.18±1.05	4.50±0.93
GAS	5.44±0.68	5.19±0.64	4.69±0.97	3.98±0.97
Journey TRAK	/	/	6.53±1.06	10.87±0.84
Relative IF	11.10±0.51	19.89±0.50	6.80±0.77	14.66±0.70
Renorm. IF	11.01±0.50	18.67±0.51	6.74±0.82	13.24±0.71
TRAK	11.28±0.47	20.02±0.47	7.02±0.89	14.71±0.70
D-TRAK	22.83±0.51	28.69±0.44	16.84±0.54	21.47±0.48
DAS	**29.38±0.51**	**33.79±0.23**	**28.73±0.49**	**30.68±0.31**

consistently outperforms existing methods. Besides, we also conduct experiment to compare DAS and D-TRAK on the entire ArtBench-10 dataset to further validate DAS's performance on large-scale and high-resolution dataset, presenting in Table 4.8.

Compared to D-TRAK, DAS shows substantial improvements. On a validation set utilizing 100 timesteps, DAS achieves improvements of +9.33% on CIFAR-2, +8.39% on ArtBench-2,

TABLE 4.8. LDS (%) on ArtBench-10 with timesteps (10 or 100)

Results on CelebA				
Method	Validation		Generation	
	10	100	10	100
D-TRAK	10.57	12.74	14.16	15.23
DAS	18.78	20.39	18.06	18.67

and +5.1% on CelebA. In the generation set, the gains continue with +5.01% on CIFAR-2, +5.78% on ArtBench-2, and +9.21% on CelebA. Notably, DAS also achieves significant improvements on larger datasets like CIFAR10 and ArtBench5, outperforming D-TRAK by +12.67% and +10.21% on validation sets and +7.58% and +8.92% on generation samples.

Other methods generally underperform on larger datasets such as ArtBench5 and CIFAR10 compared to smaller datasets like CIFAR2 and ArtBench2. Conversely, our method performs better on ArtBench5 than on ArtBench2. Remarkably, our findings suggest that while with more timesteps for calculating gradients generally leads to a better approximation of the expectation $\mathbb{E}_t$, DAS, employing only a 10-timestep computation budget, still outperforms D-TRAK, which uses a 100-timestep budget in most cases, which underscores the effectiveness of our approach. Additionally, the modest improvements on CIFAR10 and CelebA may be attributed to the LDS setup for these datasets, which employs only one random seed per subset for training a model, whereas other datasets utilize three random seeds, potentially leading to inaccuracies in LDS evaluation. Another observation is that DAS performs better on the validation set than on the generation set. This could indicate that the quality of generated images may have a significant impact on data attribution performance. However, further investigation is needed to validate this hypothesis.

4.7 Counter Factual Visualization Evaluation

To further validate the effectiveness of DAS and assess its faithfulness, we conduct a counterfactual visualization experiment. Using attribution methods including TRAK, D-TRAK, and DAS, we identify the top-1,000 positive influencers for 60 generated images from ArtBench-2

and CIFAR-2. For this process, 100 timesteps are sampled, and a projection dimension of $k = 32,768$ is used to detect the top influencers. The identified training samples are then removed from the dataset, and the model is retrained. As a baseline, a separate experiment is performed where 1,000 randomly selected training images are removed prior to retraining. The retrained models are then used to regenerate the images with the same random seeds, and the differences between the original and counterfactual images are evaluated using the pixel-wise L^2-Distance and CLIP cosine similarity.

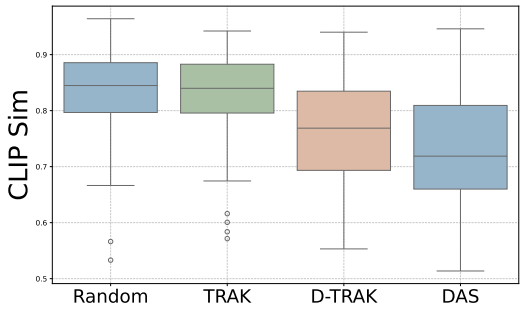
The results indicate that DAS leads to the largest changes in the generated images compared to other methods. For the L^2-Distance, D-TRAK achieves values of 8.97 and 187.61 for CIFAR-2 and ArtBench-2, respectively, while TRAK's values are 5.90 and 168.37. In contrast, DAS results in higher values of 10.58 and 203.76, reflecting its stronger impact on image generation. In terms of CLIP cosine similarity, DAS achieves median similarities of 0.83 and 0.71 for ArtBench-2 and CIFAR-2, respectively, which are lower than TRAK's values of 0.94 and 0.84 and D-TRAK's values of 0.88 and 0.77. These results demonstrate that DAS captures the influence of training data more effectively, causing the largest deviations in both pixel space and semantic similarity. The result is reported in Figure 4.5 with boxplot.

An illustration of this experiment is provided in Figure 4.4, showing that removing influencers identified by DAS results in the most significant differences from the original images. This experiment not only reinforces the limitations of LDS evaluation highlighted by [39] but also underscores the robustness of DAS in accurately identifying influential training samples and quantifying their impact on generated outputs.

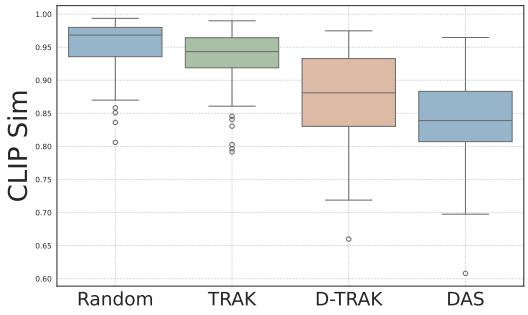
4.8 Ablation Studies

We conduct additional ablation studies to evaluate the performance differences between D-TRAK and DAS. In this section, CIFAR-2 serves as our primary setting. Further details on these settings are available in Section 4.1. We establish the corresponding LDS benchmarks as outlined in Section 4.6.

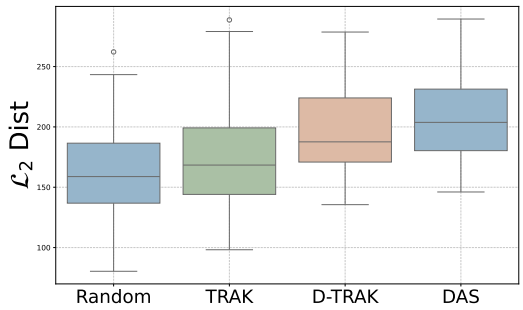
FIGURE 4.4. We conduct an visualization experiment to explore DAS effectiveness described in Sec 4.7. Removing the influential samples identified by DAS produces the most significant differences in the generated images after retraining the model. DAS is the most effective methods for attribution.



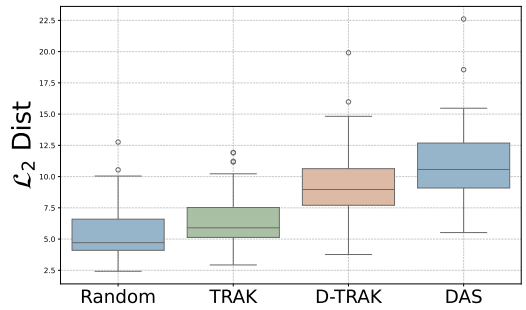
(a) ArtBench2-CLIP



(b) CIFAR2-CLIP



(c) ArtBench2-L^2 Dist.



(d) CIFAR2-L^2 Dist.

FIGURE 4.5. Boxplots of counterfactual evaluation on CIFAR2 and ArtBench2. We assess the impact of removing the 1,000 highest-scoring training samples and retraining the model using Random, TRAK, D-TRAK, and DAS. The evaluation metrics include pixel-wise L^2-Distance and CLIP cosine similarity between 60 generated samples and the corresponding images generated by the retrained models, sampled from the same random seed.

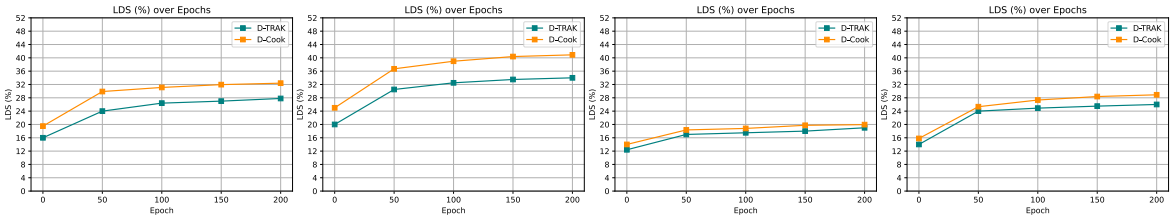


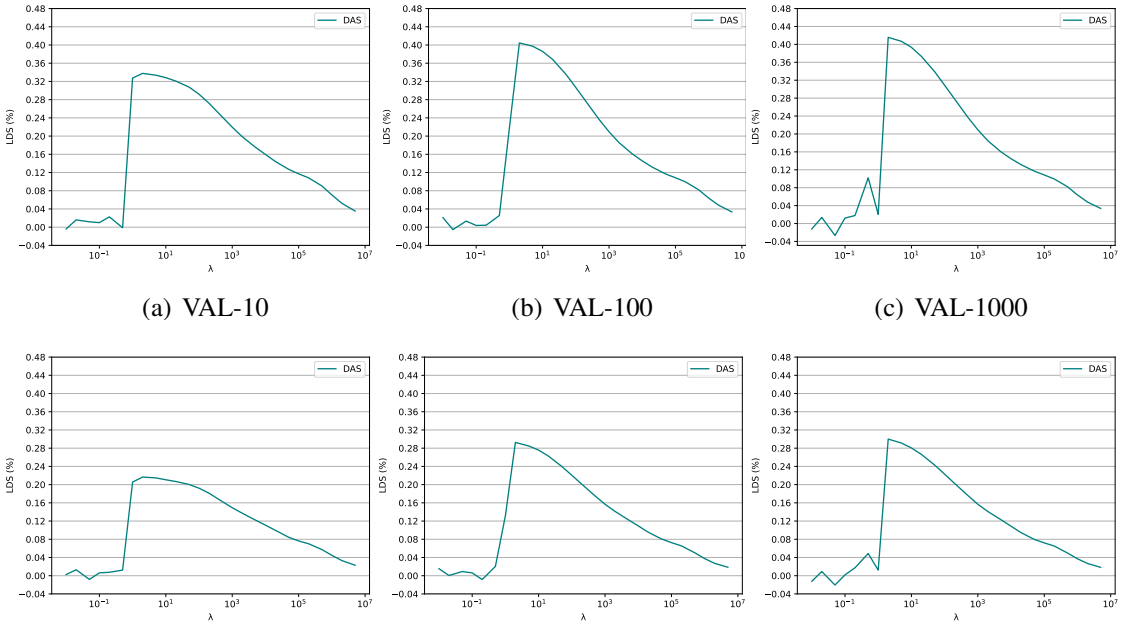
FIGURE 4.6. The LDS(%) on CIFAR-2 varies across different checkpoints. We analyze the data using 10 and 100 timesteps, evenly spaced within the interval $[1, T]$, to approximate the expectation $\mathbb{E}_t$. At each sampled timestep, we introduce one standard Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ to approximate the expectation $\mathbb{E}_\epsilon$. We set the projection dimension $k = 32768$.

4.8.1 Checkpoint selection

Following the approach outlined by [20], we investigated the impact of utilizing different model checkpoints for gradient computation. As depicted in Figures 4.6, our method achieves the highest LDS when utilizing the final checkpoint. This finding suggests that the later stages of model training provide the most accurate reflections of data influence, aligning gradients more closely with the ultimate model performance. Determining the optimal checkpoint for achieving the best LDS score requires multiple attributions to be computed, which significantly increases the computational expense. Additionally, in many practical scenarios, access may be limited exclusively to the final model checkpoint. This constraint highlights the importance of developing efficient methods that can deliver precise attributions even when earlier checkpoints are not available.

4.8.2 Value of λ

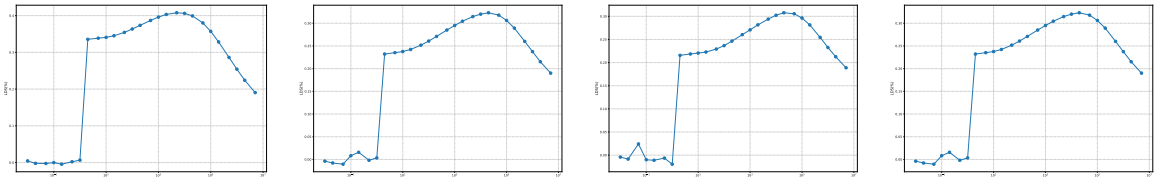
In our computation of the inverse of the Hessian matrix within the DAS framework, we incorporate the regularization parameter λ, as recommended by [62], to ensure numerical stability and effective regularization. Traditionally, λ is set to a value close to zero; however, in our experiments, a larger λ proved necessary. This is because we use the generalized Gauss-Newton (GGN) matrix to approximate the Hessian in the computation of DAS. Unlike the Hessian, the GGN is positive semi-definite (PSD), meaning it does not model negative curvature in any direction.



(a) VAL-10 (b) VAL-100 (c) VAL-1000

(d) GEN-10 (e) GEN-100 (f) GEN-1000

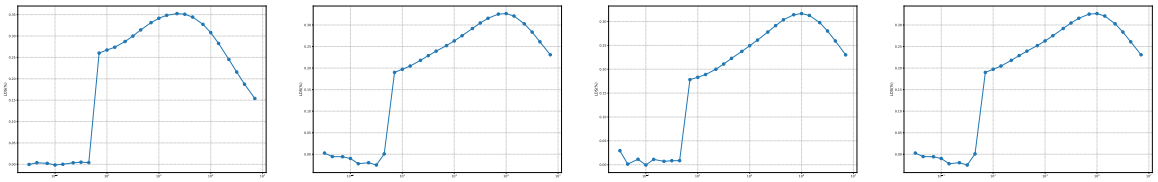
FIGURE 4.7. LDS (%) on CIFAR-2 under different λ. We consider 10, 100, and 1000 timesteps selected to be evenly spaced within the interval $[1, T]$, which are used to approximate the expectation $\mathbb{E}_t$. We set $k = 4096$.



(a) VAL-10 (b) VAL-100 (c) GEN-10 (d) GEN-100

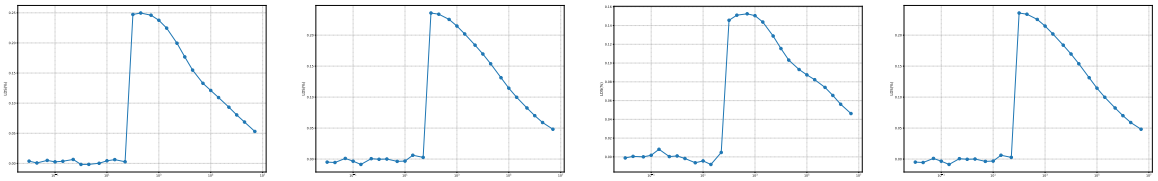
FIGURE 4.8. LDS (%) on ArtBench-2 under different λ. We consider 10 and 100 timesteps selected to be evenly spaced within the interval $[1, T]$, which are used to approximate the expectation $\mathbb{E}_t$. We set $k = 32768$.

The main issue with negative curvature is that the quadratic model predicts unbounded improvement in the objective when moving in those directions. Without certain techniques, minimizing the quadratic model results in infinitely large updates along these directions. To address this, several methods have been proposed, such as the damping technique discussed in [63].



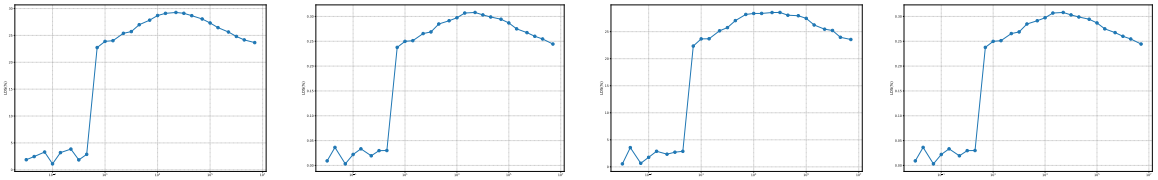
(a) VAL-10 (b) VAL-100 (c) GEN-10 (d) GEN-100

FIGURE 4.9. LDS (%) on ArtBench-5 under different λ. We consider 10 and 100 timesteps selected to be evenly spaced within the interval $[1, T]$, which are used to approximate the expectation $\mathbb{E}_t$. We set $k = 32768$.



(a) VAL-10 (b) VAL-100 (c) GEN-10 (d) GEN-100

FIGURE 4.10. LDS (%) on CIFAR-10 under different λ. We consider 10 and 100 timesteps selected to be evenly spaced within the interval $[1, T]$, which are used to approximate the expectation $\mathbb{E}_t$. We set $k = 32768$.



(a) VAL-10 (b) VAL-100 (c) GEN-10 (d) GEN-100

FIGURE 4.11. LDS (%) on CelebA under different λ. We consider 10 and 100 timesteps selected to be evenly spaced within the interval $[1, T]$, which are used to approximate the expectation $\mathbb{E}_t$. We set $k = 32768$.

In our paper, we adopt the linear damping technique λI used in [22, 35], which has proven effective on diffusion models. We show how λ influence the LDS result in Figure 4.7, Figure 4.8, Figure 4.9, Figure 4.10 and Figure 4.11.

4.9 Limitations

While our proposed Diffusion Attribution Score (DAS) showcases notable improvements in data attribution for diffusion models, several limitations warrant attention. Firstly, although DAS reduces the computational load compared to traditional methods, it still demands significant resources due to the requirement to train multiple models and compute extensive gradients. This poses challenges particularly for large-scale models and expansive datasets. Secondly, the current implementation of DAS is tailored primarily to image generation tasks. Its effectiveness and applicability to other forms of generative models, such as those for generating text or audio, remain untested and may not directly translate. Furthermore, DAS operates under the assumption that the influence of individual training samples is additive. This simplification may not accurately reflect the complex interactions and dependencies that can exist between samples within the training data.

CHAPTER 5

Conclusion

In this paper, we introduce the Diffusion Attribution Score (DAS) to address the existing gap in data attribution methodologies for generative models. We conducted a comprehensive theoretical analysis to elucidate the inherent challenges in applying TRAK to diffusion models. Subsequently, we derived DAS theoretically based on the properties of diffusion models for attributing data throughout the entire generation process. We also discuss strategies to accelerate computations to extend DAS to large-scale diffusion models. Our extensive experimental evaluations on datasets such as CIFAR, CelebA, and ArtBench demonstrate that DAS consistently surpasses existing baselines in terms of Linear Datamodeling Score evaluation. This paper underscores the crucial role of data attribution in ensuring transparency in the use of diffusion models, especially when dealing with copyrighted or sensitive content. Looking forward, our future work aims to extend DAS to other generative models and real-world applications to further ascertain its effectiveness and applicability.

Bibliography

[1] J. Lin, L. Tao, M. Dong, and C. Xu, "Diffusion attribution score: Evaluating training data influence in diffusion model," 2024.

[2] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, 2020.

[3] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in *International Conference on Learning Representations*, 2021.

[4] C. Saharia, W. Chan, H. Chang, C. Lee, J. Ho, T. Salimans, D. Fleet, and M. Norouzi, "Palette: Image-to-image diffusion models," in *ACM SIGGRAPH 2022 conference proceedings*, 2022.

[5] A. Hertz, R. Mokady, J. Tenenbaum, K. Aberman, Y. Pritch, and D. Cohen-or, "Prompt-to-prompt image editing with cross-attention control," in *The Eleventh International Conference on Learning Representations*, 2023.

[6] X. Li, J. Thickstun, I. Gulrajani, P. S. Liang, and T. B. Hashimoto, "Diffusion-lm improves controllable text generation," *Advances in Neural Information Processing Systems*, 2022.

[7] J. Ho, W. Chan, C. Saharia, J. Whang, R. Gao, A. Gritsenko, D. P. Kingma, B. Poole, M. Norouzi, D. J. Fleet, *et al.*, "Imagen video: High definition video generation with diffusion models," *arXiv preprint arXiv:2210.02303*, 2022.

[8] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022.

[9] P. W. Koh and P. Liang, "Understanding black-box predictions via influence functions," in *International conference on machine learning*, 2017.

[10] C.-K. Yeh, J. Kim, I. E.-H. Yen, and P. K. Ravikumar, "Representer point selection for explaining deep neural networks," *Advances in neural information processing systems*, 2018.

[11] A. Ilyas, S. M. Park, L. Engstrom, G. Leclerc, and A. Madry, "Datamodels: Predicting predictions from training data," in *ICML*, 2022.

[12] R. Khanna, B. Kim, J. Ghosh, and S. Koyejo, "Interpreting black box predictions using fisher kernels," in *The 22nd International Conference on Artificial Intelligence and Statistics*, 2019.

[13] R. Jia, F. Wu, X. Sun, J. Xu, D. Dao, B. Kailkhura, C. Zhang, B. Li, and D. Song, "Scalability vs. utility: Do we have to sacrifice one for the other in data importance quantification?," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[14] Z. Liu, H. Ding, H. Zhong, W. Li, J. Dai, and C. He, "Influence selection for active learning," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021.

[15] Z. Kong and K. Chaudhuri, "Understanding instance-based interpretability of variational auto-encoders," *Advances in Neural Information Processing Systems*, 2021.

[16] N. Terashita, H. Ohashi, Y. Nonaka, and T. Kanemaru, "Influence estimation for generative adversarial networks," in *International Conference on Learning Representations*, 2021.

[17] L. S. Shapley *et al.*, "A value for n-person games," *Machine Learning*, 1953.

[18] A. Ghorbani and J. Zou, "Data shapley: Equitable valuation of data for machine learning," in *International conference on machine learning*, 2019.

[19] V. Feldman and C. Zhang, "What neural networks memorize and why: Discovering the long tail via influence estimation," *Advances in Neural Information Processing Systems*, 2020.

[20] G. Pruthi, F. Liu, S. Kale, and M. Sundararajan, "Estimating training data influence by tracing gradient descent," *Advances in Neural Information Processing Systems*, 2020.

[21] S. M. Park, K. Georgiev, A. Ilyas, G. Leclerc, and A. Madry, "Trak: Attributing model behavior at scale," in *International Conference on Machine Learning*, 2023.

[22] X. Zheng, T. Pang, C. Du, J. Jiang, and M. Lin, "Intriguing properties of data attribution on diffusion models," in *The Twelfth International Conference on Learning Representations*, 2024.

[23] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[24] D. P. Kingma, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[25] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[26] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," *Advances in neural information processing systems*, 2021.

[27] R. Jia, D. Dao, B. Wang, F. A. Hubis, N. Hynes, N. M. Gürel, B. Li, C. Zhang, D. Song, and C. J. Spanos, "Towards efficient data valuation based on the shapley value," in *The 22nd International Conference on Artificial Intelligence and Statistics*, 2019.

[28] A. Schioppa, P. Zablotskaia, D. Vilar, and A. Sokolov, "Scaling up influence functions," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[29] S. Basu, P. Pope, and S. Feizi, "Influence functions in deep learning are fragile," in *International Conference on Learning Representations*, 2021.

[30] E. Akyurek, T. Bolukbasi, F. Liu, B. Xiong, I. Tenney, J. Andreas, and K. Guu, "Towards tracing knowledge in language models back to the training data," in *Findings of the Association for Computational Linguistics: EMNLP 2022*, Association for Computational Linguistics, 2022.

[31] W. Johnson and J. Lindenstrauss, "Extensions of lipschitz maps into a hilbert space," *Contemporary Mathematics*, 1984.

[32] Z. Dai and D. K. Gifford, "Training data attribution for diffusion models," 2023.

[33] S.-Y. Wang, A. A. Efros, J.-Y. Zhu, and R. Zhang, "Evaluating data attribution for text-to-image models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[34] J. Brokman, O. Hofman, R. Vainshtein, A. Giloni, T. Shimizu, I. Singh, O. Rachmil, A. Zolfi, A. Shabtai, Y. Unno, *et al.*, "Montrage: Monitoring training for attribution of generative diffusion models," in *European Conference on Computer Vision*, pp. 1–17, Springer, 2025.

[35] K. Georgiev, J. Vendrow, H. Salman, S. M. Park, and A. Madry, "The journey, not the destination: How data guides diffusion models," in *Arxiv preprint arXiv:2312.06205*, 2023.

[36] Y. Kwon, E. Wu, K. Wu, and J. Zou, "Datainf: Efficiently estimating data influence in loRA-tuned LLMs and diffusion models," in *The Twelfth International Conference on Learning Representations*, 2024.

[37] H. Shah, S. M. Park, A. Ilyas, and A. Madry, "Modeldiff: A framework for comparing learning algorithms," in *International Conference on Machine Learning*, 2023.

[38] J. Lin, A. Zhang, M. Lécuyer, J. Li, A. Panda, and S. Sen, "Measuring the effect of training data on deep learning predictions via randomized experiments," in *International Conference on Machine Learning*, 2022.

[39] Y. Hu, P. Hu, H. Zhao, and J. Ma, "Most influential subset selection: Challenges, promises, and beyond," in *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[40] L. Zhang, A. Rao, and M. Agrawala, "Adding conditional control to text-to-image diffusion models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[41] N. Carlini, J. Hayes, M. Nasr, M. Jagielski, V. Sehwag, F. Tramer, B. Balle, D. Ippolito, and E. Wallace, "Extracting training data from diffusion models," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023.

[42] G. Somepalli, V. Singla, M. Goldblum, J. Geiping, and T. Goldstein, "Diffusion art or digital forgery? investigating data replication in diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[43] S. Malladi, A. Wettig, D. Yu, D. Chen, and S. Arora, "A kernel-based view of language model fine-tuning," in *International Conference on Machine Learning*, 2023.

[44] X. Ma, G. Fang, and X. Wang, "Deepcache: Accelerating diffusion models for free," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15762–15772, 2024.

[45] E. J. Hu, yelong shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-rank adaptation of large language models," in *International Conference on Learning Representations*, 2022.

[46] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*, 2021.

[47] A. Krizhevsky, "Learning multiple layers of features from tiny images," *Master's thesis, University of Tront*, 2009.

[48] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *International Conference on Learning Representations*, 2021.

[49] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2019.

[50] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of the IEEE international conference on computer vision*, 2015.

[51] P. Liao, X. Li, X. Liu, and K. Keutzer, "The artbench dataset: Benchmarking generative models with artworks," 2022.

[52] J. Ho and T. Salimans, "Classifier-free diffusion guidance," in *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

[53] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, 2017.

[54] M. T. Ribeiro, S. Singh, and C. Guestrin, "Model-agnostic interpretability of machine learning," 2016.

[55] Z. Hammoudeh and D. Lowd, "Training data influence analysis and estimation: a survey," *Machine Learning*, 2024.

[56] R. D. Cook, "Detection of influential observation in linear regression," *Technometrics*, 1977.

[57] Y. Chen, B. Li, H. Yu, P. Wu, and C. Miao, "Hydra: Hypergradient data relevance analysis for interpreting deep neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

[58] G. Charpiat, N. Girard, L. Felardos, and Y. Tarabalka, "Input similarity from the neural network perspective," *Advances in Neural Information Processing Systems*, 2019.

[59] Z. Hammoudeh and D. Lowd, "Identifying a training-set attack's target using renormalized influence estimation," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022.

[60] E. Barshan, M.-E. Brunet, and G. K. Dziugaite, "Relatif: Identifying explanatory training samples via relative influence," in *International Conference on Artificial Intelligence and Statistics*, 2020.

[61] A. Van Den Oord, O. Vinyals, *et al.*, "Neural discrete representation learning," *Advances in neural information processing systems*, 2017.

[62] T. Hastie, "Ridge regularization: An essential concept in data science," *Technometrics*, 2020.

[63] J. Martens, "New insights and perspectives on the natural gradient method," *Journal of Machine Learning Research*, 2020.