

Method and System for Identifying Influential Training Images in Diffusion Models Using Gradient-Based Attribution

TECHNICAL FIELD

[0001] The present invention generally relates to a method and system for data attribution and, more particularly relates to a method and system for identifying influential training images in diffusion models using gradient-based attribution.

BACKGROUND

[0002] Diffusion models have become powerful tools for generating high-quality outputs, such as images and videos. These models work by progressively refining noisy images over multiple steps, resulting in high-quality outputs. However, as these models are increasingly used in various applications, it is important to understand the influence of individual training images on the generated output. Traditional methods for data attribution struggle with the complexity and multi-step nature of diffusion processes.

[0003] Data attribution in diffusion models is challenging due to the complex, iterative nature of the diffusion process. Each step in the process contributes to the final output, making it difficult to trace the influence of individual training data points.

[0004] To address potential issue of identifying the attribution for the generated output, there is a clear need to improve the method of data attribution within the diffusion model.

[0005] Furthermore, other desirable features and characteristics will become apparent from the subsequent detailed description and the appended claims, taken in conjunction with the accompanying drawings and this background of the disclosure.

SUMMARY OF THE INVENTION

[0006] In accordance with one aspect of the present invention, there is provided a method identifying an influential training image in a diffusion model for image

generation, comprising the steps of: training the diffusion model using a set of training images; generating one or more output images using the trained diffusion model; computing a gradient matrix for each image in the set of training images; calculating an importance score for the each image in the set of training images based on the computed gradient matrix, wherein the importance score indicates an importance of the set of training images to the one or more output image; and identifying one or more of the training images as the influential training image when the importance score exceeds a predetermined threshold.

[0007] In some embodiments, the importance score is τ_{D-TRAK} , and is calculated based on the equation of

$$\tau_{D-TRAK}(x, \mathcal{D}) = \left[\frac{1}{S} \sum_{s=1}^S \phi^s(x)^\top (\Phi_{D-TRAK}^s{}^\top \Phi_{D-TRAK}^s)^{-1} \Phi_{D-TRAK}^s{}^\top \right]$$

wherein the score τ_{D-TRAK} , represents how a training sample $x^n \in \mathcal{D}$ affects the diffusion model for the image generation, a total of S subsets are initially sampled from the training dataset $\mathcal{D}$, wherein $\Phi_{D-TRAK}^s = [\phi^s(x^1); \dots; \phi^s(x^N)]^\top$, which is the gradient matrix.

[0008] In some embodiments, wherein $\phi^s(x) = \nabla_{\theta} \mathcal{L}(x, \theta_s^*)$, and function $\mathcal{L}$ comprising any one of $\mathcal{L}_{Square}$, $\mathcal{L}_{Avg}$ or $\mathcal{L}_{p-norm}$, wherein:

- a. $\mathcal{L}_{Square}(x, \theta) = E_{t, \epsilon} [\|\epsilon_{\theta}(x_t, t)\|_2^2]$;
- b. $\mathcal{L}_{Avg}(x, \theta) = E_{t, \epsilon} [Avg(\epsilon_{\theta}(x_t, t))]$;
- c. $\mathcal{L}_{p-norm}(x, \theta) = E_{t, \epsilon} [\|\epsilon_{\theta}(x_t, t)\|_p]$;

wherein $x_t = \sqrt{\bar{\alpha}_t}x + \sqrt{1 - \bar{\alpha}_t}\epsilon$ and Avg is an average pooling operation.

[0009] In some embodiments, the gradient matrix computation is performed at different training data checkpoints.

[0010] In some embodiments, the method further comprises identifying training data checkpoints that contributed most to the generated one or more output images.

[0011] In some embodiments, the gradient matrix computation is performed at a final checkpoint.

[0012] In some embodiments, the diffusion model is a Denoising Diffusion Probabilistic Model (DDPM) or a Latent Diffusion Model (LDM).

[0013] In some embodiments, the method further comprises ranking the set of training images based on their importance scores to identify most important images.

[0014] In some embodiments, the method further comprises a verification step, which comprises: retraining the diffusion model after excluding the most important images identified in the ranking process; generating images using the retrained model; and measuring a pixel wise ℓ_2 distance or Contrastive Language-Image Pre-training (CLIP) cosine similarity between images generated by the original and retrained models to confirm an influence of the removed most important images.

[0015] In some embodiments, positive importance scores identify proponent images, and negative scores identify opponent images among the set of training images.

[0016] In some embodiments, the method further comprises displaying the identified influential image and/or its corresponding influence score.

[0017] In some embodiments, the method further comprises removing training images with low importance score, where the low importance score is below the predetermined threshold value; finetuning or retraining the diffusion model after removing the training images with low importance score.

[0018] In some embodiments, the method further comprises regenerating the one or more output image using the retrained diffusion model.

[0019] In accordance with another aspect of the present invention, there is provided a system of identifying an influential training image in a diffusion model for image generation, comprising: a processor; a memory in electronic communication with the processor; and instructions stored in the memory and executable by the processor to cause the system to: training the diffusion model using a set of training images; generating one or more output images using the trained diffusion model; computing a gradient matrix for each image in the set of training images; calculating an importance score for the each image in the set of training images based on the computed gradient matrix, wherein the importance score indicates an importance of the set of training images to the one or more output image; and identifying one or more of the training images as the influential training image when the importance score exceeds a predetermined threshold.

[0020] In some embodiments, the importance score is τ_{D-TRAK} , and is calculated based on the equation of

$$\tau_{D-TRAK}(x, \mathcal{D}) = \left[\frac{1}{S} \sum_{S=1}^S \phi^S(x)^\top (\Phi_{D-TRAK}^S{}^\top \Phi_{D-TRAK}^S)^{-1} \Phi_{D-TRAK}^S{}^\top \right]$$

wherein the score τ_{D-TRAK} , represents how a training sample $x^n \in \mathcal{D}$ affects the diffusion model for the image generation, a total of S subsets are initially sampled from the training dataset $\mathcal{D}$, wherein $\Phi_{D-TRAK}^S = [\phi^S(x^1); \dots; \phi^S(x^N)]^\top$, which is the gradient matrix.

[0021] In some embodiments, wherein $\phi^S(x) = \nabla_{\theta} \mathcal{L}(x, \theta_s^*)$, and function $\mathcal{L}$ comprising any one of $\mathcal{L}_{square}$, $\mathcal{L}_{Avg}$ or $\mathcal{L}_{p-norm}$, wherein:

- a. $\mathcal{L}_{square}(x, \theta) = E_{t, \epsilon} [\|\epsilon_{\theta}(x_t, t)\|_2^2]$;
- b. $\mathcal{L}_{Avg}(x, \theta) = E_{t, \epsilon} [Avg(\epsilon_{\theta}(x_t, t))]$;
- c. $\mathcal{L}_{p-norm}(x, \theta) = E_{t, \epsilon} [\|\epsilon_{\theta}(x_t, t)\|_p]$;
- d. wherein $x_t = \sqrt{\bar{\alpha}_t}x + \sqrt{1 - \bar{\alpha}_t}\epsilon$ and Avg is an average pooling operation.

[0022] In some embodiments, the gradient matrix computation is performed at different training data checkpoints.

[0023] In some embodiments, the gradient matrix computation is performed at a final checkpoint.

[0024] In some embodiments, the system further comprises displaying the identified influential training image and/or its corresponding influence score.

[0025] In accordance with another aspect of the present invention, there is provided an apparatus of identifying an influential training image in a diffusion model for image generation, comprising: means for training the diffusion model using a set of training images; means for generating one or more output images using the trained diffusion model; means for computing a gradient matrix for each image in the set of training images; means for calculating an importance score for the each image in the set of training images based on the computed gradient matrix, wherein the importance score indicates an importance of the set of training images to the one or more output image; and means for identifying one or more of the training images as the influential training image when the importance score exceeds a predetermined threshold.

[0026] In accordance with another aspect of the present invention, there is provided a computer-readable storage medium, on which a computer program is stored, wherein the computer program, when executed in a computer, causes the computer to perform the method of any one of the methods discussed hereinabove.

[0027] It should be understood that the embodiments described herein are not exhaustive and that additional features and variations of the invention may be incorporated. Various other advantages and novel features of the invention will become apparent from the following detailed description when considered in conjunction with the accompanying drawings.

[0028] The various embodiments are not necessarily mutually exclusive, as some embodiments can be combined with one or more other embodiments to form new embodiments. Embodiments described in the context of one of the systems or methods are analogously valid for the other systems or methods. Similarly, embodiments described in the context of a method are analogously valid for a system, an apparatus or a computer program, and vice-versa.

[0029] Features that are described in the context of an embodiment may correspondingly be applicable to the other embodiments, even if not explicitly described in these other embodiments. Furthermore, additions and/or combinations and/or alternatives as described for a feature in the context of an embodiment may correspondingly be applicable to the same or similar feature in the other embodiments.

BRIEF DESCRIPTION OF THE DRAWINGS

[0030] The disclosure will be better understood with reference to the detailed description when considered in conjunction with the non-limiting examples and the accompanying drawings, in which:

[0031] FIG. 1 is a system for identifying the influential training images in diffusion models according to an embodiment of the invention.

[0032] Fig. 2 is a block diagram of an electronic device for identifying the influential training images according to an embodiment of the invention.

[0033] Fig. 3 is a flowchart of a method for identifying the influential training images in diffusion models according to an embodiment of the invention.

[0034] Fig. 4 a flowchart of a method for utilizing data attribution in diffusion models to create counterfactual scenarios.

[0035] Fig. 5a is a table of Latent Diffusion Score (LDS) values using the data attribution method according to an embodiment of the invention.

[0036] Fig. 5b is the Latent Diffusion Score (LDS) values on the generation set of CIFAR-2 using checkpoints of different epochs using the data attribution method according to an embodiment of the invention.

[0037] Fig. 6 is the boxplots of counterfactual evaluation on CIFAR-2 and ArtBench-2 using the data attribution method according to an embodiment of the invention.

[0038] Fig. 7 is the counterfactual visualization on (7a, 7b) ArtBench-2 using the data attribution method according to an embodiment of the invention.

DETAILED DESCRIPTION

[0039] In the following detailed description, reference is made to the accompanying drawings, which form a part hereof. The illustrative embodiments described in the detailed description, drawings and claims are not meant to be limiting. Other embodiments can be utilized, and other changes can be made, without departing from the spirit or scope of the subject matter presented herein. Unless specified otherwise, the terms “comprising”, “comprise”, “including” and “include” used herein, and grammatical variants thereof, are intended to represent “open” or “inclusive” language such that they include recited elements but also permit inclusion of additional, un-recited elements.

[0040] FIG. 1 is a system for identifying influential training images in diffusion models according to an embodiment of the invention.

[0041] In the context of generative models, such as diffusion models, it is crucial to accurately attribute the generated images to the corresponding training images. This attribution serves the purpose of appropriately assigning credits or valuations, as well as safeguarding copyright.

[0042] This architecture diagram outlines a system designed to identify the most influential training images in diffusion models used for image generation. The system traces the model outputs back to the training data, determining which inputs have the most significant impact on the generated output images.

[0043] The system 103 receives the input training images 101. The input images are supplied to the diffusion model of the system 103. The diffusion model of the system 103 generates output images 102.

[0044] The data attribution module of the system 103 calculates gradients, assigns importance scores, and/or ranks the training images 101 based on their influence on the output image 102. The influential training images 104 are identified after the data attribution.

[0045] The system 103 further refines the training image dataset 101 based on the analysis of influential training images and prepares it for potential retraining.

[0046] It should be understood by a person skilled in the art that the described invention is applicable to various types of images, including but not limited to static images, video etc. The scope of the invention should not be constrained by the specific examples provided but should encompass all variations and adaptations that fall within the broader principles and innovative aspects described.

[0047] Fig. 2 is a block diagram of an electronic device for identifying the influential training images according to an embodiment of the invention.

[0048] In accordance with an example embodiment, the system 200 may include a data diffusion module 210, a data attribution module 220, a refine module 206, a display module 207, one or more memories 208 and one or more processors 209.

[0049] The data diffusion module 210 further comprises a data training module 201 and an image generation module 202. The diffusion model 210 takes the input training images and processes them through a series of diffusion steps to generate the final output image.

[0050] The data training module 201 is responsible for collecting and preprocessing the input training images. These images serve as the foundational data used by the diffusion model 210. The input training images are prepared and collected to the diffusion model 210 for processing.

[0051] In some embodiments, in the preprocessing step for image generation, the input images are prepared to ensure consistency and quality, which is vital for the effectiveness of the diffusion model. This process typically involves normalizing the images to a uniform scale, resizing them to match the model's input requirements, and applying data augmentation techniques such as flipping, rotating, or color adjustments to enhance the model's ability to generalize from the data. This thorough preprocessing ensures that the diffusion model starts with well-prepared data, leading to better and more reliable output images.

[0052] The image generation module 202 is the module that generates images using the diffusion model, either the original or the retrained one. This is the core model responsible for generating the output images from the input training images of the diffusion model. The diffusion model might be a Denoising Diffusion Probabilistic Model (DDPM) or any other similar model designed to generate images.

[0053] After generating the output image, the output image is then passed to either the user and/or the data attribution module 220.

[0054] The data attribution module 220 is central to identifying the most influential training images. It traces the generated output image back to the training data to determine which images had the most significant impact. It includes a gradient computation module 203, an importance score module 204 and/or a ranking module 205.

[0055] Gradient computation module 203 is the module that computes gradient values for each training image. These gradients represent how sensitive the output image is to changes in the training image.

[0056] In some embodiments, the gradient computation is implemented using a faster custom CUDA kernel. A custom CUDA kernel is a piece of code written specifically for NVIDIA GPUs to perform certain computations more efficiently. By writing a custom CUDA kernel, developers can directly control the GPU hardware, optimizing the execution of specific tasks to maximize performance. This can result in significantly faster computation compared to using pre-built functions.

[0057] For importance score calculation module 204, it calculates an importance score for each training image based on the computed gradients. It determines if the

image has a high importance score above a certain threshold. Images with importance scores above a certain threshold are considered highly influential.

[0058] In some embodiments, the system 200 further comprises a ranking module 205, it ranks the training images based on their importance scores to identify the most influential images. This module outputs a list of ranked training images based on their importance scores.

[0059] The data refine module 206, this module manages the training data based on the feedback from the data attribution module 220. It may involve retraining the model by removing or emphasizing certain training images based on their importance scores.

[0060] The refined training data can be sent back to the data attribution model 220 for further iterations or sent to the data diffusion module 210 to generate the images.

[0061] Display module 207 displays the final output images generated by the diffusion module.

[0062] In some embodiments, the importance scores of the input training images are also displayed together with the input training images.

[0063] In some embodiments, the system includes a memory 208. It stores the training images along with their importance scores for further analysis or future use. The memory 208 may include one or more non-transitory computer-readable storage media that may be non-transitory. The memory 208 may further include a high-speed random access memory and a non-volatile memory, such as one or more magnetic disk storage devices or flash storage devices. In some embodiments, a non-transitory computer-readable storage medium in the memory 208 is configured to store at least one piece of program code, the at least one piece of program code being configured to be executed by the processor 209 to implement the data diffusion and data attribution provided in the method embodiments of the invention.

[0064] In some embodiments, the user equipment includes a processor 209. The processor 209 can be implemented using various technologies and architectures designed to fulfil the described functions. It may be realized as a general-purpose

processor, content addressable memory, digital signal processor, application-specific integrated circuit, field-programmable gate array, programmable logic device, discrete gate or transistor logic, discrete hardware components, or a combination thereof. Additionally, a processor can take the form of a microprocessor, controller, microcontroller, or state machine.

[0065] Furthermore, the processor 209 may be realized through a combination of computing devices, such as combining a digital signal processor with a microprocessor, utilizing multiple microprocessors, integrating one or more microprocessors with a digital signal processor core, or employing any other suitable configuration.

[0066] In some embodiments, the processor 209 has 64 CPU cores and NVIDIA A100 GPUs each with 40GB of memory.

[0067] A person skilled in the art will recognize that the structure depicted in FIG. 2 is not limited to system 200. The system may include additional or fewer components than those illustrated, some components may be combined, or alternative component configurations may be employed.

[0068] Fig. 3 is a flowchart of a method for identifying the influential training images in diffusion models according to an embodiment of the invention. As a preliminary matter, it should be understood that steps of the method 300 are not necessarily limiting, and that steps can be added, omitted, and/or performed simultaneously without departing from the scope of the appended claims. It should be appreciated that the method 300 may include any number of additional or alternative tasks and that the method 300 may be incorporated into a more comprehensive procedure or process having additional functionality not described in detail herein. Moreover, one or more of the tasks shown in FIG. 3 could be omitted from an embodiment of the method 300 as long as the intended overall functionality remains intact. It should also be understood that the illustrated method 300 can be stopped at any time. The method 300 is computer-implemented in that various tasks or steps that are performed in connection with the method 300 may be performed by software, hardware, firmware, or any combination thereof.

[0069] The present invention primarily focuses on discrete-time diffusion models, specifically DDPMs and latent diffusion models (LDMs) that serve as the foundation of Stable Diffusion.

[0070] Data attribution in diffusion models is challenging due to the complex, iterative nature of the diffusion process. Each step in the process contributes to the final output, therefore attributing the generated images back to specific training data is challenging because each step in the diffusion process subtly alters the image, and the cumulative effect makes it difficult to trace the origin of specific image features. The current invention is related to a data attribution method to track the influence of the training data more accurately called Diffusion-Tracing with the Randomly projected After Kernel (D-TRAK).

[0071] The D-TRAK method is designed to tackle these challenges by focusing on each step in the diffusion process and quantifying the influence of each training image on the final generated output. The D-TRAK method addresses these challenges by decomposing the diffusion process into individual steps and attributing the output of each step to specific training data points. The method involves the following key steps:

[0072] At step 301, it trains the diffusion model. It gathers the input training data and trains the diffusion model using the collected data. During training, the model learns to map noisy inputs to their original, noise-free forms, thereby generating realistic outputs such as images or videos.

[0073] At step 302, it generates the output images based in the input training images by the diffusion model. Diffusion models, such as DDPMs, generate images by iteratively refining noisy images. The process begins with an initial noise image and gradually denoises it through a series of steps until a high-quality image is produced. The process involves multiple steps, each adding or removing noise, leading to a final generated output.

[0074] At step 303, it computes the gradient for each training image to assess its influence on the generated output for each data checkpoint. To understand the influence of each training image on the generated output, it computes the gradient of the output image concerning each training input. The gradient indicates how much a

small change in a training image would affect the generated output. The importance of a training image is then determined by the magnitude and direction of this gradient.

[0075] At step 304, it calculates the importance score of the input training image. In some embodiment, if the importance score assigned to a training image exceeds a certain threshold value, such as zero, it is classified as an important image. Each training image is assigned an importance score, where positive scores indicate proponents, and negative scores indicate opponents. The importance score for each training image is calculated to reflect its influence on the output image. Positive scores (proponents) signify that the training image supports or contributes positively to the generation of the output image, whereas negative scores (opponents) signify that the training image opposes or detracts from the generation of the output image.

[0076] It should be understood by a person skilled in the art that the threshold value for determining the important image can be set to different levels based on the specific requirements and objectives of the method. The techniques and methods disclosed herein are not limited to a fixed threshold and can be adapted to accommodate varying thresholds as needed to optimize performance for different applications.

[0077] At step 305, the method identifies the top influential images based on their importance scores, providing valuable insights for model interpretability and data management. Based on the calculated importance scores, it identifies the top training images that have the most significant influence on the generated output. This identification process is crucial for understanding model behavior, enhancing interpretability, and managing the ethical implications of machine generated content.

[0078] In some embodiments, the diffusion process is broken down into individual steps or different data checkpoints, each of which is analyzed separately. The computation of the gradient matrix is performed at different training data points to evaluate their contribution to the generated output images. The influence of each training data check point is measured using gradient-based methods, allowing for fine-grained attribution at each step. An importance score is calculated for each training data checkpoint, reflecting its contribution to the final output. It is crucial for

identifying which training data checkpoints have the most significant influence on the output of the diffusion model. By analyzing the gradient matrix at these different points, the method allows for a more granular understanding of how each data point affects the overall model performance.

[0079] In some embodiments, the method focuses on the final checkpoint data. To reduce computational costs, the method focuses on the final checkpoint data, which represents the state of the model after training has converged. By analyzing the final checkpoint, the method provides accurate attribution results without the need for repeated computations across all training steps. As shown in Fig. 5b, the final checkpoint consistently achieves the best performance when using the D-TRAK method. Hence, by focusing on final checkpoint data, the method reduces computational overhead while maintaining accurate attribution.

[0080] In some embodiments, the step of data attribution can also be used to optimize diffusion models by identifying which parts of the data are contributing positively or negatively to the final output. This insight can help in finetuning the model or debugging issues related to poor generation quality.

[0081] In some embodiments, the diffusion model is finetuned by identifying and removing training images with low importance scores. The importance score is calculated for each training image, and those images with scores below the predetermined threshold value are considered to have minimal influence on the desired output. By removing these low-scoring images from the training dataset, the diffusion model is retrained, allowing for a refined adjustment that optimizes the model's ability to generate more relevant and accurate outputs.

[0082] In some embodiments, the output image is regenerated after the removal of low-importance training images. Once these low-scoring images are excluded from the training dataset, the diffusion model undergoes a retraining process. Following this retraining, new output images are generated, resulting in a more optimized and targeted generation process.

[0083] Therefore, for the current invention, it provides clear identification of training images that influence the generated output, crucial for image generation transparency and accountability. It offers a robust method for identifying influential

training images in diffusion models, particularly for image generation tasks. By leveraging gradient-based attribution and importance scoring, it provides a transparent and cost-effective approach to understanding and managing the impact of training data on machine generated content.

[0084] This invention has significant implications for fields like artificial intelligence (AI) ethics, copyright protection, and model interpretability, as it enhances the ability to trace and assign credit to the specific data used in generating AI outputs. The invention could be particularly valuable in ensuring that contributors to large datasets are appropriately compensated or credited, especially when the data is used to generate high-value or copyrighted content.

[0085] Fig. 4 a flowchart of a method for utilizing data attribution in diffusion models to create counterfactual scenarios.

[0086] Similar to other machine learning models, data attribution in diffusion models can be used to create counterfactual scenarios. For example, by tweaking certain inputs and observing changes in the output, one can understand the model's sensitivity to specific data points, leading to better model robustness. The method of creating the counterfactual scenarios involves the following key steps.

[0087] At step 401, in some embodiment, the training dataset is preprocessed to prepare it for analysis. This may involve standard data augmentation techniques, normalization, and other preprocessing methods to ensure that the data is suitable for training and evaluation.

[0088] At step 402, it removes the important images identified in Fig. 3. After the importance of each image is quantified, it allows the identification of the most important images that have a significant impact on the diffusion model's output. The identified most important images are removed from the training dataset.

[0089] At step 403, the diffusion model is then retrained using the modified dataset, which excludes the top-ranking influential images. This step is critical in creating a counterfactual scenario, as it allows the analysis of how the absence of these important images affects the model's performance.

[0090] At step 404, after the retraining, the modified diffusion model is used to generate new images. These images represent the counterfactual scenario, where the influence of the most important training images has been removed.

[0091] At step 405, the impact of the removed training images is evaluated by comparing the images generated by the original model with those generated by the retrained model. This comparison is performed using pixel-wise ℓ_2 distance metrics and Contrastive Language-Image Pre-training (CLIP) cosine similarity. These metrics provide quantitative measures of how the removal of the most important images has influenced the generated outputs, confirming their significance in the original model. A large pixel-wise ℓ_2 distance metric indicates that the regenerated image is significantly different from the input image after excluding the top influential training examples. Similarly, a low CLIP cosine similarity score reflects the same effect, showing that the regenerated image has diverged substantially from the original image due to the removal of these influential examples.

[0092] Therefore, by identifying and potentially removing the influential images to create the create counterfactual scenarios, the method allows for the refinement of training datasets, leading to improved model performance.

EXAMPLES

[0093] The system and method herein disclosed are further illustrated in the following examples, which are provided by way of illustration and are not intended to be limiting the scope of the present disclosure.

DIFFUSION MODELS

[0094] In one embodiment, the diffusion model is the DDPMs. The details of the DDPM are as follows: DDPMs process define a random variable $\mathbf{x} \in \mathcal{X}$ and define a forward diffusion process on $\mathbf{x}$ as $\mathbf{x}_{1:T} \triangleq \mathbf{x}_1, \dots, \mathbf{x}_T$ with $T \in \mathbb{N}^+$. The data distribution is $\mathbf{x} \sim q(\mathbf{x})$ and the Markov transition probability from $\mathbf{x}_{t-1}$ to $\mathbf{x}_t$ is $q(\mathbf{x}_t|\mathbf{x}_{t-1}) \triangleq \mathcal{N}(\mathbf{x}_t|\sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$, where $\mathbf{x}_0 = \mathbf{x}$ and $\beta_1, \dots, \beta_T$ correspond to a variance schedule. A notable property of DDPMs is that they can sample $\mathbf{x}_t$ at an arbitrary timestep t directly from $\mathbf{x}$, since there is

$q(\mathbf{x}_t|\mathbf{x}) = \mathcal{N}(\mathbf{x}_t|\sqrt{\bar{\alpha}_t}\mathbf{x}, (1 - \bar{\alpha}_t)\mathbf{I})$, where $\alpha_t \triangleq 1 - \beta_t$ and $\bar{\alpha}_t \triangleq \prod_{i=1}^t \alpha_i$. In some embodiment, when β_t are small, the reverse diffusion process can also be modeled by Gaussian conditionals.

[0095] Specifically, for the DDPMs framework, the reverse transition probability from $\mathbf{x}_t$ to $\mathbf{x}_{t-1}$ is written as $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}|\boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \sigma_t^2\mathbf{I})$, where $\theta \in \mathbb{R}^d$ is the model parameters and σ_t are time dependent constants that can be predefined or analytically computed. Instead of directly modeling the data prediction $\boldsymbol{\mu}_\theta$, DDPMs choose to model the noise prediction $\boldsymbol{\epsilon}_\theta$ based on the parameterization $\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\alpha_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right)$. The training objective of $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$ can be derived from optimizing the variational bound of negative log-likelihood formulated as follows:

$$\mathcal{L}_{\text{ELBO}}(\mathbf{x}; \theta) = \mathbb{E}_{\boldsymbol{\epsilon}, t} \left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)} \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|_2^2 \right], \quad (1)$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}|\mathbf{0}, \mathbf{I})$ and $t \sim \mathcal{U}([1, T])$ denotes the discrete uniform distribution between 1 and T . Let $\mathcal{D} \triangleq \{\mathbf{x}^n\}_{n=1}^N$ be a training dataset that $\mathbf{x}^n \sim q(\mathbf{x})$, then the empirical training objective on $\mathcal{D}$ can be written as $\mathcal{L}_{\text{ELBO}}(\mathcal{D}; \theta) = \frac{1}{N} \sum_{\mathbf{x}^n \in \mathcal{D}} \mathcal{L}_{\text{ELBO}}(\mathbf{x}^n, \theta)$. To benefit sample quality, DDPMs apply a simplified training objective that corresponds to a weighted variational bound and is formulated as

$$\mathcal{L}_{\text{Simple}}(\mathbf{x}; \theta) = \mathbb{E}_{\boldsymbol{\epsilon}, t} \left[\left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|_2^2 \right], \quad (2)$$

where the empirical objective on $\mathcal{D}$ is similarly written as

$$\mathcal{L}_{\text{Simple}}(\mathcal{D}; \theta) = \frac{1}{N} \sum_{\mathbf{x}^n \in \mathcal{D}} \mathcal{L}_{\text{Simple}}(\mathbf{x}^n, \theta).$$

DATA ATTRIBUTION AND EVALUATION METRICS:

[0096] Data attribution refers to the goal of tracing model outputs back to training data. Consider an ordered training set of samples $\mathcal{D} \triangleq \{\mathbf{x}^n\}_{n=1}^N$ and a model output function $\mathcal{F}(\mathbf{x}; \theta)$. A data attribution method $\tau(\mathbf{x}, \mathcal{D})$ is a function $\tau : \mathcal{X} \times \mathcal{X}^N \rightarrow \mathbb{R}^N$ that, for any sample $\mathbf{x} \in \mathcal{X}$ and a training set $\mathcal{D}$, assigns a score to each training input $\mathbf{x}^n \in \mathcal{D}$ indicating its importance to the model output $\mathcal{F}(\mathbf{x}; \theta^*(\mathcal{D}))$, where $\theta^*(\mathcal{D}) = \arg \min_{\theta} \mathcal{L}(\mathcal{D}; \theta)$.

[0097] In some embodiments, the linear data modeling score (LDS) is used to evaluate data attribution, which considers the sum of attributions as an additive proxy for $\mathcal{F}$, as a metric for evaluating data attribution methods. It is noted that there are various other metrics to evaluate data attribution methods, including but not limited to leave-one-out influences, Shapley values, and performance on auxiliary tasks. This score is a direct measure of the influence that each training image has on the final output image. A higher score indicates that the corresponding training image plays a more critical role in shaping the characteristics and quality of the generated image, making it a key contributor to the model's performance.

[0098] For the LDS evaluation method, it defines the attribution-based output prediction of the model output $\mathcal{F}(\mathbf{x}; \theta^*(\mathcal{D}'))$ as

$$g_{\tau}(\mathbf{x}, \mathcal{D}'; \mathcal{D}) \triangleq \sum_{\mathbf{x}^n \in \mathcal{D}'} \tau(\mathbf{x}, \mathcal{D})_n, \quad (3)$$

where $\mathcal{D}'$ is a subset of $\mathcal{D}$ as $\mathcal{D}' \subset \mathcal{D}$. Then the LDS metric can be constructed as follows:

Considering a training set $\mathcal{D}$, a model output function $\mathcal{F}(x; \theta)$, and a corresponding data attribution method τ . Let $\{\mathcal{D}^m\}_{m=1}^M$ be M randomly sampled subsets of the training dataset $\mathcal{D}$ that $\mathcal{D}^m \subset \mathcal{D}$, each of size $\alpha \cdot N$ for some $\alpha \in (0, 1)$. The linear datamodeling score (LDS) of a data attribution τ for a specific sample $x \in \mathcal{X}$ is given by

$$\text{LDS}(\tau, x) \triangleq \rho(\{\mathcal{F}(x; \theta^*(\mathcal{D}^m)) : m \in [M]\}, \{g_\tau(x, \mathcal{D}^m; \mathcal{D}) : m \in [M]\}), \quad (4)$$

where ρ denotes Spearman rank correlation.

[0099] In some embodiments, to counter the randomness of the training mechanism (the process of approximating $\theta^*(\mathcal{D}^m)$), for every subset $\mathcal{D}^m$, it trains three models with different random seeds and average the model output function. It reduces variance in the model's performance.

[0100] In some embodiments, a counterfactual evaluation approach is used to assess the utility of the data attribution methods. it compares the pixel-wise ℓ_2 -distance and CLIP cosine similarity of generated images, using the models trained before/after the exclusion of the highest-ranking positive influencers identified by different attribution methods. Specifically, the approach comprises the steps of retraining the models after selectively removing training images identified as the highest-ranking positive influencers; comparing the pixel-wise ℓ_2 -distance between images generated by the original and retrained models to quantify the impact of excluding influential training examples; evaluating the CLIP cosine similarity between images generated by the original and retrained models to quantify the impact of excluding influential training examples.

[0101] This process ensures that the attribution methods are thoroughly evaluated in terms of their ability to accurately identify and rank the most influential training examples that affect the output of the diffusion model.

ATTRIBUTION METHODS:

TRACING WITH THE RANDOMLY PROJECTED AFTER KERNEL (TRAK)

[0102] In some embodiments, the data attribution is implemented using the method of TRAK. The primary interface employed by attribution methods is the score $\tau(\mathbf{x}, \mathcal{D})$, which is calculated for each training input to indicate its importance to the output of interest. The influence functions are a classical concept that approximates how much an infinitesimally up-weighting of a training sample $\mathbf{x}^n \in \mathcal{D}$ affects the model output function $\mathcal{F}(\mathbf{x}; \theta^*)$, measured on a sample of interest $\mathbf{x}$. In the convex setting, the attributing score of influence function can be computed as

$\tau_{\text{IF}}(\mathbf{x}, \mathcal{D})_n = \nabla_{\theta} \mathcal{F}(\mathbf{x}; \theta^*)^{\top} \cdot \mathcal{H}_{\theta^*}^{-1} \cdot \nabla_{\theta} \mathcal{L}(\mathbf{x}^n; \theta^*)$ for $n \in [N]$, where $\mathcal{H}_{\theta^*} = \nabla_{\theta}^2 \mathcal{L}(\mathcal{D}; \theta^*)$ is the Hessian matrix at the optimal parameters θ^* .

[0103] TRAK aims to enhance the efficiency and scalability of attributing discriminative classifiers. In the TRAK algorithm, a total of S subsets denoted as $\{\mathcal{D}^s\}_{s=1}^S$ are initially sampled from the training dataset $\mathcal{D}$, where each subset has a fixed size of $\beta \cdot N$ for $\beta \in (0, 1]$. On each subset $\mathcal{D}^s$, a model is trained to obtain the parameters $\theta_s^* \in \mathbb{R}^d$ and a random projection matrix $\mathcal{P}_s$ is sampled from $\mathcal{N}(0, 1)^{d \times k}$ (typically there is $k \ll d$). Then TRAK constructs the projected gradient matrices Φ_{TRAK}^s and the weighting terms $\mathcal{Q}_{\text{TRAK}}^s$ as

$$\begin{aligned} \Phi_{\text{TRAK}}^s &= [\phi^s(\mathbf{x}^1); \dots; \phi^s(\mathbf{x}^N)]^{\top}, \text{ where } \phi^s(\mathbf{x}) = \mathcal{P}_s^{\top} \nabla_{\theta} \mathcal{F}(\mathbf{x}; \theta_s^*); \\ \mathcal{Q}_{\text{TRAK}}^s &= \text{diag}(Q^s(\mathbf{x}^1), \dots, Q^s(\mathbf{x}^N)), \text{ where } Q^s(\mathbf{x}) = \frac{\partial \mathcal{L}}{\partial \mathcal{F}}(\mathbf{x}; \theta_s^*). \end{aligned} \tag{5}$$

Finally, the attribution score $\tau_{\text{TRAK}}(\mathbf{x}, \mathcal{D})$ of TRAK is computed by

$$\tau_{\text{TRAK}}(\mathbf{x}, \mathcal{D}) = \left[\frac{1}{S} \sum_{s=1}^S \phi^s(\mathbf{x})^\top \left(\Phi_{\text{TRAK}}^s{}^\top \Phi_{\text{TRAK}}^s \right)^{-1} \Phi_{\text{TRAK}}^s{}^\top \right] \left[\frac{1}{S} \sum_{s=1}^S \mathcal{Q}_{\text{TRAK}}^s \right]. \quad (6)$$

[0104] In the discriminative classification cases, the model output function is $\mathcal{F}(\mathbf{x}; \theta) = \log(\exp(\mathcal{L}(\mathbf{x}; \theta)) - 1)$ according to the mechanism of logistic regression.

DIFFUSION- TRACING WITH THE RANDOMLY PROJECTED AFTER KERNEL (D-TRAK)

[0105] In some embodiments, the data attribution is implemented using the method of D-TRAK.

[0106] Given a DDPM trained by minimizing $\mathcal{L}(\mathcal{D}; \theta) = \mathcal{L}_{\text{Simple}}(\mathcal{D}; \theta)$, there is $\theta^*(\mathcal{D}) = \arg \min_{\theta} \mathcal{L}_{\text{Simple}}(\mathcal{D}; \theta)$ and the model output function is set to be $\mathcal{F}(\mathbf{x}; \theta) = \mathcal{L}(\mathbf{x}; \theta) = \mathcal{L}_{\text{Simple}}(\mathbf{x}, \theta)$.

[0107] In this particular configuration, when it computes the attribution score of TRAK in Eq. 6, the weighting terms $\mathcal{Q}_{\text{TRAK}}^s = \frac{\partial \mathcal{L}}{\partial \mathcal{F}} = \mathbf{I}$ become identity matrices, and the function ϕ^s is constructed as $\phi^s(\mathbf{x}) = \mathcal{P}_s^\top \nabla_{\theta} \mathcal{L}_{\text{Simple}}(\mathbf{x}, \theta_s^*)$.

[0108] In some embodiments, it considers TRAK with $\phi^s(\mathbf{x}) = \mathcal{P}_s^\top \nabla_{\theta} \mathcal{L}_{\text{Simple}}(\mathbf{x}, \theta_s^*)$ as a suitable approach for attributing DDPMs. This is particularly applicable where both $\mathcal{L}$ and $\mathcal{F}$ are $\mathcal{L}_{\text{Simple}}$.

[0109] In some embodiments, the attribution is calculated by a D-TRAK. It replaces $\phi_s(\mathbf{x})$ with alternative functions can result in higher values of LDS. Specifically, the formula is

$$\tau_{\text{D-TRAK}}(\mathbf{x}, \mathcal{D}) = \left[\frac{1}{S} \sum_{s=1}^S \phi^s(\mathbf{x})^\top \left(\Phi_{\text{D-TRAK}}^s \Phi_{\text{D-TRAK}}^{s\top} \right)^{-1} \Phi_{\text{D-TRAK}}^s \right], \quad (7)$$

where $\Phi_{\text{D-TRAK}}^s = [\phi^s(\mathbf{x}^1); \dots; \phi^s(\mathbf{x}^N)]^\top$ are the projected gradient matrices. D-TRAK allows ϕ^s to be constructed from alternative functions, rather than relying on $\mathcal{F}$ as $\phi^s(\mathbf{x}) = \mathcal{P}_s^\top \nabla_{\theta} \mathcal{F}(\mathbf{x}; \theta_s^*)$. The weighting terms are eliminated (i.e., $\mathcal{Q}_{\text{D-TRAK}}^s = \mathbf{I}$) under the assumption that $\mathcal{L}$ and $\mathcal{F}$ are the same. In addition to $\mathcal{L}_{\text{Simple}}$ and $\mathcal{L}_{\text{ELBO}}$, alternative functions of $\mathcal{L}_{\text{Square}}$, $\mathcal{L}_{\text{Avg}}$, and $\mathcal{L}_{p\text{-norm}}$ are defined to be the constructing ϕ^s in D-TRAK, formulated as

$$\mathcal{L}_{\text{Square}}(\mathbf{x}, \theta) = \mathbb{E}_{t, \epsilon} [\|\epsilon_{\theta}(\mathbf{x}_t, t)\|_2^2]; \mathcal{L}_{\text{Avg}}(\mathbf{x}, \theta) = \mathbb{E}_{t, \epsilon} [\text{Avg}(\epsilon_{\theta}(\mathbf{x}_t, t))]; \mathcal{L}_{p\text{-norm}}(\mathbf{x}, \theta) = \mathbb{E}_{t, \epsilon} [\|\epsilon_{\theta}(\mathbf{x}_t, t)\|_p],$$

where $\mathbf{x}_t = \sqrt{\alpha_t} \mathbf{x} + \sqrt{1 - \alpha_t} \epsilon$ and $\text{Avg}(\cdot) : \mathcal{X} \rightarrow \mathbb{R}$ is the average pooling operation. It instantiates $p = 1, 2, \infty$ for $\mathcal{L}_{p\text{-norm}}$.

INTERPOLATION BETWEEN $\mathcal{L}_{\text{Simple}}$ and $\mathcal{L}_{\text{Square}}$

[0110] It is noteworthy that several alternative functions, namely $\mathcal{L}_{\text{Square}}$, $\mathcal{L}_{\text{Avg}}$, $\mathcal{L}_{2\text{-norm}}$, and $\mathcal{L}_{1\text{-norm}}$, consistently outperform the seemingly reasonable choice of $\mathcal{L}_{\text{Simple}}$. To take a closer look on how these phenomena occur, $\mathcal{L}_{\text{Square}}$ is taken as an object of study, and expand the gradients of $\mathcal{L}_{\text{Simple}}$ and $\mathcal{L}_{\text{Square}}$ as

$$\nabla_{\theta} \mathcal{L}_{\text{Simple}} = \mathbb{E}_{t, \epsilon} \left[2 \cdot (\epsilon_{\theta} - \epsilon)^{\top} \nabla_{\theta} \epsilon_{\theta} \right] \text{ and } \nabla_{\theta} \mathcal{L}_{\text{Square}} = \mathbb{E}_{t, \epsilon} \left[2 \cdot \epsilon_{\theta}^{\top} \nabla_{\theta} \epsilon_{\theta} \right], \quad (8)$$

where the dependence on $\mathbf{x}$ and ϵ is omitted for the simplicity of notations. It is found that $\nabla_{\theta} \mathcal{L}_{\text{Simple}}$ and $\nabla_{\theta} \mathcal{L}_{\text{Square}}$ share the same term of $\nabla_{\theta} \epsilon_{\theta}$, and the difference is that they product $\nabla_{\theta} \epsilon_{\theta}$ with $2 \cdot (\epsilon_{\theta} - \epsilon)^{\top}$ and $2 \cdot \epsilon_{\theta}^{\top}$, respectively. It is deduced that the information of $\nabla_{\theta} \epsilon_{\theta}$ is better retained in the norm-based losses. Hence, it performs interpolation on these two loss functions and subsequently utilize the resulting function to construct ϕ^s in D-TRAK:

$$\phi^s(\mathbf{x}) = \mathcal{P}_s^{\top} \nabla_{\theta} [\eta \mathcal{L}_{\text{Square}} + (1 - \eta) (\mathcal{L}_{\text{Simple}} - \mathcal{L}_{\text{Square}})] (\mathbf{x}, \theta_s^*) = \mathbb{E}_{t, \epsilon} \left[2 \cdot (\eta \epsilon_{\theta} - (1 - \eta) \epsilon)^{\top} \nabla_{\theta} \epsilon_{\theta} \right],$$

where $\eta \in [0, 1]$ is the interpolation hyperparameter. When $\eta = 0.5$, there is $\phi^s(\mathbf{x}) = \frac{1}{2} \mathcal{P}_s^{\top} \nabla_{\theta} \mathcal{L}_{\text{Simple}}$ corresponding to TRAK (the constant factor $\frac{1}{2}$ does not affect LDS); when $\eta = 1$, there is $\phi^s(\mathbf{x}) = \mathcal{P}_s^{\top} \nabla_{\theta} \mathcal{L}_{\text{Square}}$ corresponding to D-TRAK ($\mathcal{L}_{\text{Square}}$). TRAK (i.e., $\eta = 0.5$) has the poorest performance compared to other interpolations. Moreover, as the value of η diverges further from 0.5, approaching either 0 or 1, the corresponding LDS values increase.

EXPERIMENTS

DATASETS

[0111] The input images are from three datasets including CIFAR (32×32), CelebA (64×64), and ArtBench (256×256).

[0112] In some embodiments, CIFAR (32×32) data set is used. The CIFAR-10 dataset comprises 50,000 training samples. It randomly samples 1,000 validation samples from CIFAR-10's test set for LDS evaluation. To reduce computation, a

CIFAR-2 dataset is also constructed as a subset of CIFAR-10, which consists of 5,000 training samples randomly sampled from CIFAR-10’s training samples corresponding to the “automobile” and “horse” classes, and 1,000 validation samples randomly sampled from CIFAR-10’s test set corresponding to the same two classes.

[0113] On CIFAR, the model architecture has 35.7M parameters (i.e., $d = 35.7 \times 10^6$ for $\theta \in \mathbb{R}^d$). The maximum timestep is $T = 1000$, and the linear variance schedule for the forward diffusion process is $\beta_1 = 10^{-4}$ to $\beta_T = 0.02$. The dropout rate is set to be 0.1, employ the AdamW optimizer with weight decay of 10^{-6} , and augment the data with random horizontal flips. A DDPM is trained for 200 epochs with a 128 batch size, using a cosine annealing learning rate schedule with a 0.1 fraction warmup and an initial learning rate of 10^{-4} . During inference, new images are generated utilizing the 50-step DDIM solver.

[0114] In some embodiments, CelebA (64×64) data set is used. It samples a subset of 5,000 training samples and 1,000 validation samples from the original training set and test set of CelebA, respectively, and first center crop the images to 140×140 and then resize them to 64×64.

[0115] On CelebA, it uses an unconditional DDPM implementation that is similar to the one used for CIFAR after being adapted to 64×64 resolution by slightly modifying the U-Net architecture, which has 118.8M parameters. Other hyper-parameters are identical to those employed on CIFAR.

[0116] In some embodiments, ArtBench (256×256) data set is used. ArtBench is a dataset for artwork generation. It includes 60,000 artwork images for 10 distinct artistic styles, with 5,000 training images and 1,000 testing images per style. ArtBench-2 is also constructed as a subset of ArtBench, consisting of 5,000 training/1,000 validation samples sampled from 10,000 training/2,000 test samples of the “post-impressionism” and “ukiyo-e” classes. The method is tested also test on ArtBench-5 as a subset of ArtBench, consisting of 12,500 training/1,000 validation samples sampled from 25,000 training/5,000 test samples of the “post-impressionism”, “ukiyo-e”, “romanticism”, “renaissance” and “baroque” classes.

[0117] On ArtBench, it fine-tunes the Stable Diffusion model using LoRA with the rank set to 128, which consists of 25.5M parameters. To fit the resolution ratio of

ArtBench, it uses a Stable Diffusion checkpoint that has been adapted from 512×512 to 256×256 resolution. The model is trained in a class-conditional manner where the textual prompt is simply set as “a {class} painting” such as “a romanticism painting”. It also sets the dropout rate to 0.1, employ the AdamW optimizer with weight decay of 10^{-6} , and augment the data with random horizontal flips. The model is trained for 100 epochs under a batch size of 64, using a cosine annealing learning rate schedule with 0.1 fraction warmup and an initial learning rate of 3×10^{-4} . At the inference phase, new images are sampled using the 50-step DDIM solver and a guidance scale of 7.5.

[0118] In addition to the 1,000 held-out validation samples, a set of 1,000 generated images are created for each of the aforementioned datasets. Notably, calculating LDS necessitates retraining a large number of models, which constitutes the majority of the computational cost. In contrast, several attribution methods, such as TRAK and D-TRAK, are computationally efficient and scalable to larger datasets.

Evaluating LDS for attribution methods

[0119] To conduct the LDS evaluation, it samples $M = 64$ different random subsets of the training set $\mathcal{D}$, and trains three models with different random seeds on each one of these subsets. Each subset sampled to 50% of the size of the training set, that is, it sets $\alpha = 0.5$. It then computes the linear datamodeling score for each sample of interest as the Spearman rank correlation between the model output and the attribution-based output prediction of the model output as described in Eq. 3. Especially, when computing $\mathcal{L}_{\text{Simple}}(\mathbf{x}; \theta)$ as described in Eq.2 or any sample of interest from either the validation set or generation set, it considers all the 1000 timesteps selected to be evenly spaced within the interval $[1, T]$ to approximate the expectation $\mathbb{E}_t$. For each timestep, it samples three standard Gaussian noises $\epsilon \sim \mathcal{N}(\epsilon | \mathbf{0}, \mathbf{I})$ to approximate the expectation $\mathbb{E}_\epsilon$. In some embodiment, the LDS is averaged across samples of interest from the validation set and generation set respectively.

[0120] Fig. 5a is a table of Latent Diffusion Score (LDS) values using the data attribution method according to an embodiment of the invention.

[0121] The table shows the LDS (%) values on CIFAR-2 (a subset consisting of two classes from CIFAR-10) with different constructions of $\phi^s(\mathbf{x})$. All the values of LDS are calculated with $\mathcal{F} = \mathcal{L} = \mathcal{L}_{\text{Simple}}$, and the model is a DDPM with $T = 1000$. LDS values are computed on both the validation set (original test images) and the generation set (newly generated images) with respect to the training samples. Regarding the trade-off between computational demand and efficiency, different numbers of timesteps are considered (e.g., 10, 100, and 1000) sampled from $t \sim \mathcal{U}([1, T])$ to approximate the expectation of $\mathbb{E}_t$, where these timesteps are selected to be evenly spaced within the interval $[1, T]$ (by the *arange* operation). For each sampled timestep, it samples one standard Gaussian noise $\epsilon \sim \mathcal{N}(\epsilon|\mathbf{0}, \mathbf{I})$ to approximate the expectation $\mathbb{E}_\epsilon$. The projection dimension of each $\mathcal{P}_s$ is $k = 4096$. While $\phi^s(\mathbf{x}) = \mathcal{P}_s^\top \nabla_{\theta} \mathcal{L}_{\text{Simple}}(\mathbf{x}, \theta_s^*)$ should be a reasonable design choice for attributing DDPMs, it is counter-intuitive to observe that using ϕ^s constructed by $\mathcal{L}_{\text{Square}}$, $\mathcal{L}_{\text{Avg}}$, $\mathcal{L}_{2\text{-norm}}$, and $\mathcal{L}_{1\text{-norm}}$ consistently achieve higher values of LDS. As can be seen from table, D-TRAK constructed from $\mathcal{L}_{\text{Square}}$, $\mathcal{L}_{\text{Avg}}$, $\mathcal{L}_{2\text{-norm}}$, and $\mathcal{L}_{1\text{-norm}}$ consistently outperform TRAK by a large margin.

[0122] Fig. 5b is the LDS(%) values on the generation set of CIFAR-2 using checkpoints of different epochs using the data attribution method according to an embodiment of the invention. It selects 10, 100, and 1000 timesteps evenly spaced within the interval $[1, T]$ to approximate $\mathbb{E}_t$. For each selected timestep, it samples one standard Gaussian noise to approximate $\mathbb{E}_\epsilon$. The value of k is set as $k = 32768$. The plot 510 is when the timesteps is 10. The plot 520 is when the timesteps is 100. The plot 530 is when the timesteps is 1000.

[0123] As shown in Fig. 5b, for D-TRAK, the best LDS scores are obtained at the final checkpoint. However, for TRAK, using the final checkpoint is not the best choice. Finding which checkpoint yields the best LDS score requires computing the attributions for many times, which means much more computational cost. In practice, it might also only get access to the final model.

[0124] In summary, when computing gradients, the final model checkpoint is used as default. 10 and 100 timesteps are selected to be evenly spaced within the interval $[1, T]$. For example, the selected timesteps are $\{1, 101, 201, \dots, 901\}$ when the number of timesteps is 10. For each timestep, it samples one standard Gaussian noise. The projection dimension is $k = 32768$.

COUNTERFACTUAL EVALUATION

[0125] To evaluate the faithfulness of D-TRAK more intuitively, it measures the pixel-wise ℓ_2 -distance and CLIP cosine similarity on CIFAR-2, CelebA, and ArtBench-2 datasets. These measurements are taken between images generated by the models trained both before and after excluding the top-1000 positive influencers, as identified by different attribution methods. For both D-TRAK and TRAK, it considers 100 timesteps, samples one standard Gaussian noise, and sets $k = 32768$. A control baseline called Random is considered, which randomly removes 1000 training samples before retraining.

[0126] Fig. 6 is the boxplots of counterfactual evaluation on CIFAR-2 and ArtBench-2 using the data attribution method according to an embodiment of the invention. It quantifies the impact of removing the 1,000 highest-scoring training samples and re-training according to Random, TRAK, and D-TRAK. The pixel-wise ℓ_2 -distance and CLIP cosine similarity between 60 synthesized samples are measured and corresponding images generated by the re-trained models when sampling from the same random seed.

[0127] When examining the median pixel-wise ℓ_2 distance resulting from removing-and-retraining, D-TRAK yields 15.07, in contrast to TRAK’s values of 11.02. A reverse trend is observed when evaluating the median CLIP cosine similarity, where D-TRAK obtains lower similarities of 0.896, which are notably lower than TRAK’s 0.942.

[0128] The counterfactual evaluation of D-TRAK against TRAK, as presented in Fig. 6, demonstrates that D-TRAK can better identify influential images that have a larger impact on the generated images. As shown in Fig.6, when examining the median pixel-wise ℓ_2 distance resulting from removing-and-retraining, D-TRAK yields

8.97, 15.07, and 187.61 for CIFAR-2, and ArtBench-2, respectively, in contrast to TRAK’s values of 5.90 and 168.37. D-TRAK obtains median similarities of 0.881 and 0.769 for the above three datasets respectively, which are notably lower than TRAK’s values of 0.943 and 0.840, highlighting the effectiveness of the D-TRAK method.

[0129] Fig. 7 is the counterfactual visualization on (7a, 7b) ArtBench-2 using the data attribution method according to an embodiment of the invention.

[0130] In the context of ArtBench-2 datasets, it compares the original generated samples to those generated from the same random seed with the models trained after the exclusion of the top 1,000 positive influencers identified by different attribution methods including Random, TRAK, and D-TRAK. For both D-TRAK and TRAK, it considers 100 timesteps selected to be evenly spaced within the interval $[1, T]$, samples one standard Gaussian noise, and sets $k = 32768$ when computing the gradients.

[0131] As shown in Fig. 7, the results suggest that the method D-TRAK can better identify influential images that have a significant impact on the target image. Take Fig. 7a as an example, the model retrained after removing training samples based on Random and TRAK still generates a human face image, while the one corresponding to D-TRAK generates an image looks similar to a table.

[0132] In some embodiments, it classifies the training samples that have a positive influence score as proponents and samples that have a negative value of influence score as opponents. For samples of interest, the training samples corresponding to the most positive and negative attribution scores are visualized. It is observed that D-TRAK consistently finds proponents visually more similar to the target samples.

[0133] In some embodiments, the training samples that have the highest self-influence scores on CIFAR-2 and ArtBench-2 are also visualized. The self-influence of $\mathbf{x}^n$ is computed as $\tau_{\text{D-TRAK}}(\mathbf{x}^n, \mathcal{D})_n$. When computing the self-influence scores, it considers 100 timesteps selected to be evenly spaced within the interval $[1, T]$, and set $k = 32768$. The identified highly self-influenced samples usually look atypical or

unique, while low self-influence ones have similar samples in the training set. Especially, on CIFAR-2, highly self-influenced samples are visually high-contrast.

[0134] For the current invention, any suitable programming language can be used to implement the routines of particular embodiments including C, C++, Java, assembly language, etc. Different programming techniques can be employed such as procedural or object oriented. The routines can execute on a single processing device or multiple processors.

[0135] Particular embodiments may be implemented in a computer-readable storage medium (also referred to as a machine-readable storage medium) for use by or in connection with the instruction execution system, apparatus, system, or device. Particular embodiments can be implemented in the form of control logic in software or hardware or a combination of both. The control logic, when executed by one or more processors, may be operable to perform that which is described in particular embodiments.

[0136] While various aspects and embodiments have been disclosed herein, it will be apparent that various other modifications and adaptations of the invention will be apparent to the person skilled in the art after reading the foregoing disclosure without departing from the spirit and scope of the invention and it is intended that all such modifications and adaptations come within the scope of the appended claims. The various aspects and embodiments disclosed herein are for purposes of illustration and are not intended to be limiting, with the true scope and spirit of the invention being indicated by the appended claims.

Claims

1. A method of identifying an influential training image in a diffusion model for image generation, comprising the steps of:

training the diffusion model using a set of training images;

generating one or more output images using the trained diffusion model;

computing a gradient matrix for each image in the set of training images;

calculating an importance score for the each image in the set of training images based on the computed gradient matrix, wherein the importance score indicates an importance of the set of training images to the one or more output image; and

identifying one or more of the training images as the influential training image when the importance score exceeds a predetermined threshold.

2. The method of claim 1, wherein the importance score is τ_{D-TRAK} , and is calculated based on the equation of

$$\tau_{D-TRAK}(x, \mathcal{D}) = \left[\frac{1}{S} \sum_{s=1}^S \phi^s(x)^\top (\Phi_{D-TRAK}^s{}^\top \Phi_{D-TRAK}^s)^{-1} \Phi_{D-TRAK}^s{}^\top \right]$$

wherein the score τ_{D-TRAK} , represents how a training sample $x^n \in \mathcal{D}$ affects the diffusion model for the image generation, a total of S subsets are initially sampled from the training dataset $\mathcal{D}$, wherein $\Phi_{D-TRAK}^s = [\phi^s(x^1); \dots; \phi^s(x^N)]^\top$, which is the gradient matrix.

3. The method of claim 2, wherein $\phi^s(x) = \nabla_{\theta} \mathcal{L}(x, \theta_s^*)$, and function of $\mathcal{L}$ comprising any one of $\mathcal{L}_{Square}$, $\mathcal{L}_{Avg}$ or $\mathcal{L}_{p-norm}$, wherein:

$$\mathcal{L}_{Square}(x, \theta) = E_{t, \epsilon} [\|\epsilon_{\theta}(x_t, t)\|_2^2];$$

$$\mathcal{L}_{Avg}(x, \theta) = E_{t, \epsilon} [Avg (\epsilon_{\theta}(x_t, t))];$$

$$\mathcal{L}_{p-norm}(x, \theta) = E_{t, \epsilon} [\|\epsilon_{\theta}(x_t, t)\|_p];$$

wherein $x_t = \sqrt{\bar{\alpha}_t}x + \sqrt{1 - \bar{\alpha}_t}\epsilon$ and Avg is an average pooling operation.

4. The method of any of claims 1 to 3, wherein the gradient matrix computation is performed at different training data checkpoints.
5. The method of any of claims 1 to 4, further comprising identifying training data checkpoints that contributed most to the generated one or more output images.
6. The method of any of claims 1 to 5, wherein the gradient matrix computation is performed at a final checkpoint.
7. The method of any of claims 1 to 6, wherein the diffusion model is a Denoising Diffusion Probabilistic Model (DDPM) or a Latent Diffusion Model (LDM).
8. The method of any of claims 1 to 7, further comprising ranking the set of training images based on their importance scores to identify most important images.
9. The method of claim 8, further comprising a verification step, which comprises:
retraining the diffusion model after removing the most important images identified in the ranking process;
generating images using the retrained model; and
measuring a pixel wise ℓ_2 distance or Contrastive Language-Image Pre-training (CLIP) cosine similarity between images generated by the original and retrained models to confirm an influence of the removed most important images.
10. The method of any of claims 1 to 9, wherein positive importance scores identify proponent images, and negative scores identify opponent images among the set of training images.
11. The method of any of claims 1-10, further comprising displaying the identified influential image and/or its corresponding importance score.
12. The method of any of claims 1-11, further comprising
removing training images with low importance score, where the low importance score is below the predetermined threshold value;
finetuning or retraining the diffusion model after removing the training images with low importance score.
13. The method of claim 12, further comprising
regenerating the one or more output image using the retrained diffusion model
14. A system of identifying an influential training image in a diffusion model for image generation, comprising:

a processor;
a memory in electronic communication with the processor; and
instructions stored in the memory and executable by the processor to cause the system to:

- train the diffusion model using a set of training images;
- generate one or more output images using the trained diffusion model;
- compute a gradient matrix for each image in the set of training images;
- calculate an importance score for the each image in the set of training images based on the computed gradient matrix, wherein the importance score indicates an importance of the set of training images to the one or more output image; and
- identify one or more of the training images as the influential training image when the importance score exceeds a predetermined threshold.

15. The system of claim 14, wherein the importance score is τ_{D-TRAK} , and is calculated based on the equation of

$$\tau_{D-TRAK}(x, \mathcal{D}) = \left[\frac{1}{S} \sum_{S=1}^S \phi^S(x)^\top (\Phi_{D-TRAK}^S{}^\top \Phi_{D-TRAK}^S)^{-1} \Phi_{D-TRAK}^S{}^\top \right]$$

wherein the score τ_{D-TRAK} , represents how a training sample $x^n \in \mathcal{D}$ affects the diffusion model for the image generation, a total of S subsets are initially sampled from the training dataset $\mathcal{D}$, wherein $\Phi_{D-TRAK}^S = [\phi^S(x^1); \dots; \phi^S(x^N)]^\top$, which is the gradient matrix.

16. The system of claim 15, wherein $\phi^S(x) = \nabla_\theta \mathcal{L}(x, \theta_s^*)$, and function of $\mathcal{L}$ comprising any of $\mathcal{L}_{Square}$, $\mathcal{L}_{Avg}$ or $\mathcal{L}_{p-norm}$, wherein:

$$\mathcal{L}_{Square}(x, \theta) = E_{t, \epsilon} [\|\epsilon_\theta(x_t, t)\|_2^2];$$

$$\mathcal{L}_{Avg}(x, \theta) = E_{t, \epsilon} [Avg(\epsilon_\theta(x_t, t))];$$

$$\mathcal{L}_{p-norm}(x, \theta) = E_{t, \epsilon} [\|\epsilon_\theta(x_t, t)\|_p];$$

wherein $x_t = \sqrt{\bar{\alpha}_t}x + \sqrt{1 - \bar{\alpha}_t}\epsilon$ and Avg is an average pooling operation.

17. The system of any of claims 14-16, wherein the gradient matrix computation is performed at a final checkpoint.
18. The system of any of claims 14-17, further comprising displaying the identified influential training image and/or its corresponding influence score.
19. An apparatus of identifying an influential training image in a diffusion model for image generation, comprising:
- means for training the diffusion model using a set of training images;
 - means for generating one or more output images using the trained diffusion model;
 - means for computing a gradient matrix for each image in the set of training images;
 - means for calculating an importance score for the each image in the set of training images based on the computed gradient matrix, wherein the importance score indicates an importance of the set of training images to the one or more output image; and
 - means for identifying one or more of the training images as the influential training image when the importance score exceeds a predetermined threshold.
20. A computer-readable storage medium, on which a computer program is stored, wherein the computer program, when executed in a computer, causes the computer to perform the method of any of claims 1-13.