A SURVEY OF DATA ATTRIBUTION: METHODS, APPLICATIONS, AND EVALUATION IN THE ERA OF GENERATIVE AI

WORKING PAPER

Junwei Deng[1*] **Yuzheng Hu**[1*†] **Pingbang Hu**[1*] **Ting-Wei Li**[1*] **Shixuan Liu**[1*] **Jiachen T. Wang**[2]
Dan Ley[3] **Qirun Dai**[4] **Benhao Huang**[5] **Jin Huang**[6] **Cathy Jiao**[5] **Hoang Anh Just**[7] **Yijun Pan**[6]
Jingyan Shen[8] **Yiwen Tu**[9] **Weiyi Wang**[6] **Xinhe Wang**[5] **Shichang Zhang**[3] **Shiyuan Zhang**[1]
Ruoxi Jia[7] **Himabindu Lakkaraju**[3] **Hao Peng**[1] **Weijing Tang**[5] **Chenyan Xiong**[5] **Jieyu Zhao**[10]
Hanghang Tong[1] **Han Zhao**[1] **Jiaqi W. Ma**[1†]
[1]University of Illinois Urbana-Champaign [2]Princeton University [3]Harvard University
[4]University of Chicago [5]Carnegie Mellon University [6]University of Michigan [7]Virginia Tech
[8]New York University [9]University of California San Diego [10]University of Southern California

September 6, 2025

ABSTRACT

Training data is the fuel of modern artificial intelligence (AI), fundamentally shaping the capabilities, limitations, and biases of AI systems. The emergence of large-scale generative models has elevated the importance of understanding how data influences their behaviors, bringing the field of *data attribution* to the forefront. This survey provides a comprehensive overview of data attribution, covering its methods, applications, and evaluation protocols, with a particular emphasis on the challenges and opportunities arising in the era of generative AI. We start by introducing a conceptual framework for attribution centered on three core questions: *what to attribute* (model behaviors), *attribute to what* (training entities), and *how to attribute* (influence measures). Within this framework, we systematically review major attribution approaches, including those based on influence functions, weighted marginal contributions, training dynamics, and simulators. We then examine key applications of data attribution, such as data selection, fact tracing, adversarial attacks and defenses, and the emerging data economy. Finally, we critically assess common evaluation criteria, including the quality of counterfactual predictions, utility in downstream tasks, and computational efficiency. We conclude with a forward-looking perspective on the future of data attribution, highlighting key open challenges and promising directions for future research.

Keywords Data Attribution · Generative AI

1 Introduction

Training data is the cornerstone of modern artificial intelligence (AI), significantly shaping model behaviors through its scale, quality, and composition. These data characteristics profoundly influence critical aspects of AI models, including performance (Kaplan et al., 2020), robustness (Gowal et al., 2021), privacy (Carlini et al., 2021), and safety (Qi et al., 2024). Consequently, precisely understanding how variations in training data affect model outcomes has become increasingly crucial. This has led to the emergence of *data attribution*, a field focused on systematically quantifying the contribution of individual training data samples or data subsets a model's predictions, behaviors, or internal states. Since the seminal work by Koh and Liang (2017), which introduced influence functions (Hampel, 1974; Hampel et al., 2011) as a practical tool for data attribution, there has been a surge of new developments in this field, offering more scalable and precise techniques (Ghorbani and Zou, 2019; Pruthi et al., 2020; Hara et al., 2019; Park et al., 2023; Wang and Jia, 2023a; Wang et al., 2024h).

[*]Co-first authors.
[†]Corresponding to {yh46, jiaqima}@illinois.edu.

The advent of large-scale generative AI models, such as diffusion models (Ho et al., 2020b) and large language models (LLMs) (Brown et al., 2020), has placed an even greater emphasis on the importance of training data, in turn posing new demands on data attribution. These demands arise from several interconnected challenges and opportunities unique to the generative paradigm. First, the immense scale of these models, often comprising billions of parameters trained on vast datasets, renders many traditional attribution methods computationally infeasible and necessitates the development of highly efficient and scalable techniques (Schioppa et al., 2022; Kwon et al., 2024; Choe et al., 2024). Second, the complexity of generative tasks, from autoregressive language modeling to diffusion-based image generation, expands the scope of attribution itself. This requires new definitions for both the model behaviors to attribute (e.g., generation quality, factual recall (Akyürek et al., 2022), or safety (Cohen and Giryes, 2024)) and the training entities to which we attribute them (e.g., individual tokens (Yeh et al., 2022), data domains (Xie et al., 2023), or abstract concepts (Cohen-Wang et al., 2024) beyond single examples). Third, generative AI creates novel and critical applications for data attribution. These include verifying the factual grounding of generated text (Akyürek et al., 2022), ensuring compliance with copyright and privacy regulations by tracing generated content (Deng et al., 2024b), understanding emergent phenomena like in-context learning (Cohen-Wang et al., 2024), and diagnosing complex failure modes such as hallucination (Lin et al., 2024b). Finally, evaluating attribution methods in this new landscape is itself a major challenge. Traditional evaluation strategies that depend on repeated model retraining are hindered by prohibitive computational costs, while the inherent stochasticity of model training further complicates the reliability of assessments (Nguyen et al., 2023).

This survey provides a comprehensive overview of the field of data attribution, examining its **methods**, **applications**, and **evaluation**, with a special focus on its evolution in the context of generative AI. For methods (Section 2), we first introduce a conceptual framework for understanding data attribution, organized around three fundamental questions: "What to attribute?" (*model behaviors*), "Attribute to what?" (*training entities*), and "How to attribute?" (*influence measures*). This framework provides a foundation for understanding data attribution methods. Building on this framework, we systematically review state-of-the-art attribution methods, which we categorize according to the principles of their influence measures (i.e., "How to attribute?"). For applications (Section 3), we explore the diverse and emerging application scenarios of data attribution in generative AI, including *data selection*, *fact tracing*, *adversarial attacks and defenses*, and *data economy*. For evaluation (Section 4), we critically assess the major types of evaluation criteria from the literature, including *counterfactual prediction of model behavior*, *utility in downstream tasks*, and *computational costs*, and highlight their strengths and limitations. Finally , we conclude the survey with a discussion of the survey scope (Section 5), and a forward-looking perspective on the future of data attribution in the era of generative AI (Section 6). Figure 1 provides a visual roadmap of these components and their interconnections.

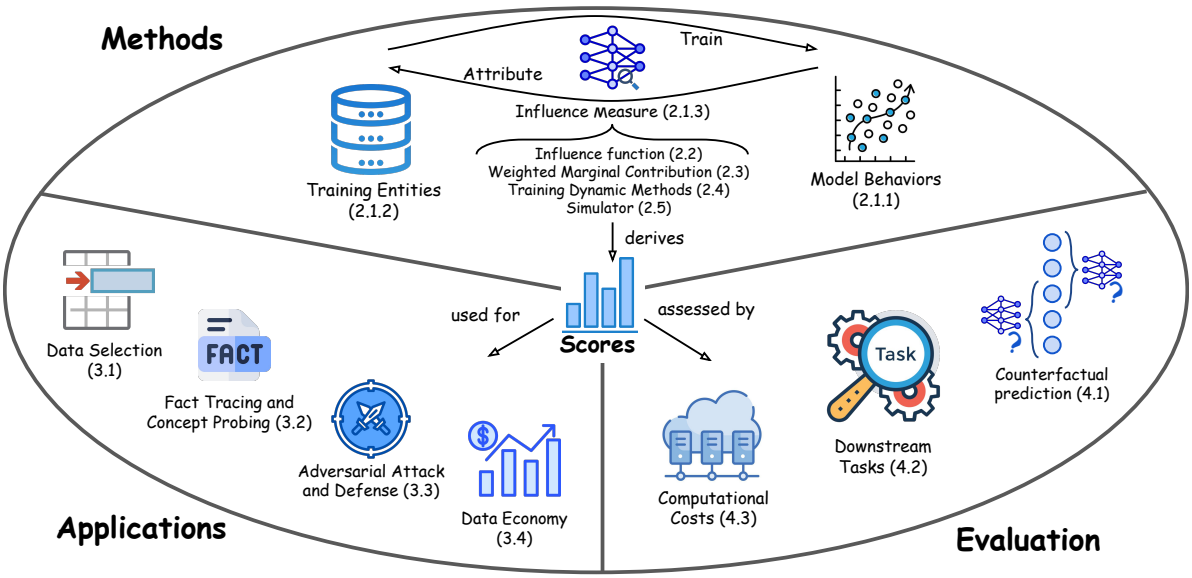

Figure 1: Overview of the key components in this survey, comprising methods, applications, and evaluation. The *methods* section presents a high-level conceptual framework consisting of training entities, influence measures, and model behaviors. The *applications* section explores various sub-applications that can leverage data attribution. The *evaluation* section categorizes existing evaluation methods into distinct types. Corresponding section numbers are indicated in parentheses.

2 Methods

This section provides a systematic study of training data attribution methods. We begin by introducing the fundamental concepts of data attribution. We then review mainstream attribution methods, discussing their definitions, underlying principles, and challenges in scaling to large generative models. Finally, we compare these approaches in terms of efficiency, attribution quality, and applicability, highlighting their trade-offs and practical considerations.

Definition. *Data attribution* measures the influence (Section 2.1.3) of each training entity (Section 2.1.2) on the model behavior (Section 2.1.1).

2.1 Foundations of Data Attribution

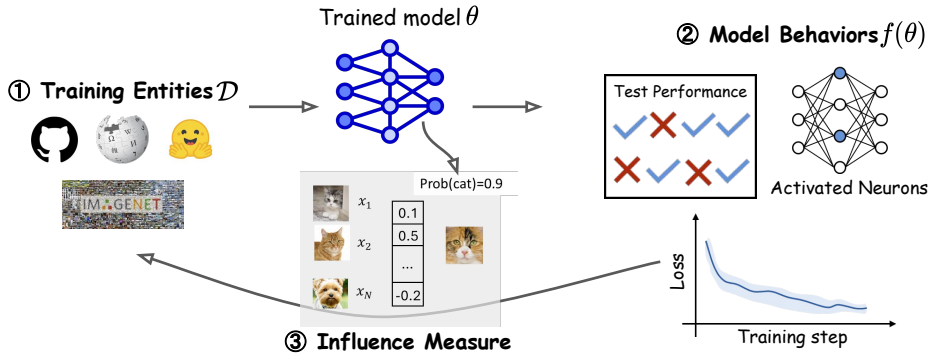

Figure 2: An overview of the three fundamental components in data attribution. Models are trained on *training entities*, which leads to various types of *model behaviors*. Data attribution quantifies the *influence measure* of each training entity on these behaviors.

We decompose data attribution into three fundamental components: **model behavior**, which represents the outcome to be attributed; **training entity**, the source responsible for these behaviors; and **influence measure**, which quantifies the contribution of training entities to the observed model behavior. See Figure 2 for an overview. We systematically review each component, providing the necessary foundation for understanding different training data attribution methods.

2.1.1 Model Behavior: What to Attribute?

The definition of model behavior centers around the question: **what to attribute?** This involves identifying the *measurable* effects of a model, typically tied to a well-defined *target function*. These effects can be *static*, such as a trained model's prediction or loss evaluated on a specific test sample or an entire test set, or *dynamic*, capturing how these predictions or losses evolve throughout the training process.

Traditionally, data attribution has been primarily applied to standard supervised learning tasks, such as classification (Koh and Liang, 2017; Yeh et al., 2018; Pruthi et al., 2020). However, the emergence of advanced generative models, including diffusion models (Ho et al., 2020b; Nichol and Dhariwal, 2021; Song and Ermon, 2019; Song et al., 2021b,c) and large language models (LLMs) (Brown et al., 2020; Ouyang et al., 2022; Achiam et al., 2023; Touvron et al., 2023a,b; Dubey et al., 2024), has significantly expanded the scope of model behavior. These models, particularly due to their strong multi-task capabilities (Radford et al., 2019), introduce richer and more diverse definitions of model behaviors, giving rise to novel attribution challenges (e.g., fact tracing (Akyürek et al., 2022; Chang et al., 2025), also see Section 3 for more examples). To provide a structured overview, Table 1 summarizes commonly used target functions across different generative models, along with their representative references.

2.1.2 Training Entity: Attribute to What?

In parallel, the definition of training entity addresses the question: **attribute to what?** At its core, this focuses on identifying the *source* that drives the model's behavior, providing insights into its underlying mechanisms. The vast majority of the literature attributes model behavior to *individual training samples*, which is the most natural and

Table 1: Commonly used target functions for different types of generative models.

Model Type	Target Functions	Representative References
Decoder-Only LLMs	Negative log likelihood	Choe et al. (2024)
	Multi-class Margin	Park et al. (2023)
Diffusion Models	DDPM loss	Georgiev et al. (2023)
	Norm-based noise prediction	Zheng et al. (2023)
	Noise prediction	Lin et al. (2025)
Other Models	CLIP loss	Park et al. (2023)

widely applicable formulation. Accordingly, our discussion in Sections 2.2 to 2.6 will primarily consider attribution to individual samples unless otherwise noted.

While individual sample attribution is the standard, some studies explore alternative formulations. In LLMs, attribution can be performed at a finer granularity, such as *token-level* attribution (Lin et al., 2024b), where the influence of individual tokens is assessed. Another approach is *concept-level* attribution (Cohen-Wang et al., 2024), which aims to identify broader semantic structures such as key phrases, document sections, or contextual passages that are influential. Conversely, attribution can be extended to *groups of samples* (Koh et al., 2019), often under the assumption of *additivity* (Hu et al., 2024a). Additionally, in settings involving multiple data distributions, some methods attribute model behavior to entire datasets or training domains (Wang et al., 2023a; Liu et al., 2025). We summarize different training entity formulations in the literature, along with their representative references, in Table 2.

Table 2: Granularity of training entities.

Granularity	Description	Representative References
Token	Attribution to individual tokens	Yeh et al. (2022); Lin et al. (2024b)
Concept	Attribution to broader semantic structures	Cohen-Wang et al. (2024); Chuang et al. (2025)
Sample	Attribution to individual samples	Grosse et al. (2023); Park et al. (2023), Wang et al. (2024g,h, 2025b)
Group	Attribution to groups of training samples	Koh et al. (2019); Basu et al. (2020), Hu et al. (2024a); Ley et al. (2024)
Domain	Attribution to training domains	Xie et al. (2023); Wang et al. (2023a), Kang et al. (2024); Liu et al. (2025)

2.1.3 Influence Measure: How to Attribute?

An influence measure serves as the *channel* between training entities and model behavior by providing a systematic method to quantify the impact of each training entity on the observed behavior. It answers the question of **how to attribute**, establishing a computational framework for linking sources to outcomes.

Influence measures typically assign *scores* to training entities, most commonly individual training samples. These scores reflect the importance of each training entity in shaping the model's behavior and are typically interpreted in two ways: 1) **Magnitude**: A larger absolute value indicates a stronger influence; 2) **Sign**: A positive score suggests a training entity reinforces the model's behavior, whereas a negative score indicates an opposing effect.

Within the extensive literature on data attribution, we identify the following primary categories of influence measures:

- **Influence function** (Section 2.2): Use first-order approximations of model parameters to efficiently approximate the effect of removing a single training sample (i.e., leave-one-out (LOO)) without requiring full retraining. The seminal work in this line is Koh and Liang (2017). Additionally TRAK (Park et al., 2023) is introduced as a variant of influence function.
- **Weighted marginal contribution** (Section 2.3): Quantify sample influence based on its expected marginal contribution when added to subsets of the training data, where each subset is assigned a specific weight. Methods such as Data Shapley (Ghorbani and Zou, 2019; Jia et al., 2019a) and Data Banzhaf (Wang and Jia, 2023a) fall into this category.

- **Training dynamic methods** (Section 2.4): Leverage intermediate model checkpoints to analyze how training samples dynamically affect model behaviors along the training trajectory. Two prominent methods in this category are TracIn (Pruthi et al., 2020) and SGD-influence (Hara et al., 2019).
- **Simulator** (Section 2.5): Use surrogate models (e.g., Datamodels (Ilyas et al., 2022)) to estimate the counterfactual effect of model training. We will also discuss the connections between this category and weighted marginal contribution methods.
- **Others** (Section 2.6): Methods that do not fit into the above categories, such as those based on unlearning (Wang et al., 2024h) or similarity (Yang et al., 2025).

Influence measures form the core of data attribution, providing a quantitative framework for assessing the impact of training entities and are typically agnostic to the choice of target functions. In Sections 2.2 to 2.6, we analyze specific methods in detail. For each category, we first review early approaches that established the core methodologies. We then examine the challenges of scaling these methods to large-scale generative models. Finally, we discuss recent advancements aimed at addressing these challenges and highlight open questions that remain in the field.

2.2 Category: Influence Function

The first category of methods focuses on the *influence function* (`IF`), a tool originating in robust statistics (Hampel, 1974; Hampel et al., 2011) and is closely related to cross validation (Debruyne et al., 2008) and jackknife (Giordano et al., 2019b). `IF` was first introduced to deep learning in the seminal work by Koh and Liang (2017). It provides a principled way to understand how changes in individual training data points affect model parameters and predictions.

Notation. Let $z = (x, y)$ denote a data point, where x is the input and y is the corresponding label. The loss incurred on a single data point is denoted by $\ell(z; \theta)$, where $\theta \in \mathbb{R}^p$ represents the model parameters. Given a training set $D = \{z_i\}_{i=1}^n$, the empirical risk is defined as $\mathcal{L}(\theta) = \frac{1}{n}\sum_{i=1}^n \ell(z_i; \theta)$, and the optimal model parameters are obtained by $\hat{\theta} = \arg\min_\theta \mathcal{L}(\theta)$. We denote θ_{-z} as the optimal model parameters obtained by $\arg\min_\theta \frac{1}{n}\sum_{i=1}^n \ell(z_i; \theta) - \frac{1}{n}\ell(z; \theta)$. We denote the Hessian of the empirical risk evaluated at θ by $H_\theta = \nabla_\theta^2 \mathcal{L}(\theta)$.

Derivation of `IF`. The key idea in Koh and Liang (2017) is to approximate the change in model parameters when a single training point z is removed, i.e., $\hat{\theta}_{-z} - \hat{\theta}$, without the need for expensive retraining. To formalize this idea, consider a perturbed empirical risk that upweights z by a small amount ϵ:

$$\mathcal{L}_{\epsilon,z}(\theta) = \frac{1}{n}\sum_{i=1}^n \ell(z_i; \theta) + \epsilon\, \ell(z; \theta), \tag{1}$$

and let $\hat{\theta}_{\epsilon,z}$ denote the minimizer of this perturbed objective. Here, $\epsilon = -\frac{1}{n}$ corresponds to leave-one-out (LOO) (i.e., $\hat{\theta}_{-z}$) while $\epsilon = 0$ corresponds to the original model parameters $\hat{\theta}$. When ℓ is strictly convex, we can apply the implicit function theorem (Krantz and Parks, 2002) and compute the local perturbation of $\hat{\theta}_{\epsilon,z}$ around $\epsilon = 0$ as

$$\left.\frac{\mathrm{d}\hat{\theta}_{\epsilon,z}}{\mathrm{d}\epsilon}\right|_{\epsilon=0} = -H_{\hat{\theta}}^{-1} \nabla_\theta \ell(z; \hat{\theta}).$$

Building on this formula, `IF` leverages a first-order Taylor expansion to approximate the change in parameters as

$$\hat{\theta}_{-z} - \hat{\theta} \approx \frac{1}{n} H_{\hat{\theta}}^{-1} \nabla_\theta \ell(z; \hat{\theta}). \tag{2}$$

In practice, we are often interested not only in how the parameters change but also in the impact of these changes on a specific quantity. As discussed in Section 2.1.1, this is typically called the target function, denoted as $f(\theta)$. For example, a common target function is the loss at a test point, $f(\theta) = \ell(z_{\text{test}}; \theta)$. By applying the chain rule, we propagate the parameter change's effect as follows

$$f(\hat{\theta}_{-z}) - f(\hat{\theta}) \approx \nabla_\theta f(\hat{\theta})^\top (\hat{\theta}_{-z} - \hat{\theta}) \approx \frac{1}{n} \nabla_\theta f(\hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_\theta \ell(z; \hat{\theta}). \tag{3}$$

This expression quantifies the influence of the training point z on the target function f, encapsulating the central idea of *`IF` as a first-order approximation to LOO retraining*.

Challenges. Koh and Liang (2017) demonstrates that, although the derivation of IF relies on the loss function being strongly convex and the model is trained to convergence, IF can be adapted to non-convex models and still perform reasonably well. However, their experiments are restricted to small-scale neural networks. When scaling IF to larger and complex models, which are prevalent in generative AI, several challenges emerge. For example, Basu et al. (2021) shows that IF is sensitive to model size and various training hyperparameters. Meanwhile, Hammoudeh and Lowd (2022) argues that IF induces a low-loss penalty, causing confidently predicted training samples to be ranked as uninfluential. Bae et al. (2022) show that, in practice, IF in deep neural networks approximates a different measure, i.e., the proximal Bregman response function, rather than exact LOO. Finally, even putting aside these methodological concerns, the computational cost and memory footprint of IF, which involves the (inverse) Hessian and gradients for all training samples, become prohibitively large as both model and dataset sizes increase.

These challenges have spurred two research directions[3] particularly relevant to generative AI: one that tackles non-convexity and non-convergence issues in large-scale models, and another that improves the scalability of IF's. In Section 2.2.1 and Section 2.2.2, we examine these research avenues respectively. Moreover, in Section 2.2.4, we examine TRAK (Park et al., 2023), a variant of IF that has recently attracted significant attention. We conclude this section by summarizing key takeaways and outlining future work in Section 2.2.5.

2.2.1 Addressing Non-convexity and Non-convergence

Non-strongly convexity and non-convergence are two key bottlenecks that limit the application of IF to large-scale models. In this section, we review existing strategies aimed at mitigating these challenges and highlight several open questions for future research. We note that trajectory-specific LOO (Hara et al., 2019; Bae et al., 2024; Wang et al., 2025b), a line of work within the category of training dynamic methods, also aims to address these challenges. We discuss it in more detail in Section 2.4.2.

Singularity of Hessian. One of the primary challenges when applying influence functions to non-strongly convex models is that the Hessian matrix may be singular or not positive-definite. In such cases, the inverse of the Hessian, which is required for calculating the IF as in Equation (2), becomes unstable or undefined. To address this issue, a common strategy, introduced by Koh and Liang (2017), is to add a damping term λI to the Hessian to make it positive-definite. This damping term can be interpreted as equivalent to adding an L_2-regularization to Equation (1), thereby constraining the model parameters to a small neighborhood around their optimal values. This constraint not only enhances numerical stability but also reinforces the validity of the first-order approximation, thereby improving IF's attribution quality (Schioppa et al., 2022). However, tuning the regularization coefficient is challenging without a clear ground truth; indeed, Zhang and Zhang (2022) demonstrates that IF's accuracy is sensitive to this choice, with too small a value significantly degrading performance.

Another widely adopted approach is to replace the Hessian with an approximation with benign properties. One such choice is the Fisher Information Matrix (FIM) (Fisher, 1922); specifically, for models trained with the negative log-likelihood objective, the FIM is equivalent to Hessian *in expectation* (a.k.a. Bartlett's second identity (Bartlett, 1953)), where the expectation is taken with respect to the model's predictive distribution. Unlike the Hessian in non-convex settings, the FIM is guaranteed to be positive semi-definite. Recent work (e.g., Teso et al., 2021; Kwon et al., 2024; Zhou et al., 2024) has replaced the vanilla Hessian with the *empirical* FIM in IF, achieving promising results. An alternative to the FIM is the Generalized Gauss-Newton (GGN) matrix (Bae et al., 2022; Grosse et al., 2023; Mlodozeniec et al., 2025), which is equivalent to the FIM in many practical cases (Martens, 2020). Both the FIM and GGN have been shown to offer a more robust and stable approximation of the Hessian, thereby enhancing the quality of attributions derived from IF. Additionally, the simpler formulation of the FIM as a sum of outer products of gradients leads to improved computational efficiency. We will discuss this further in Section 2.2.2.

Randomness from model training. Recent studies have shown that the validity of IF can be compromised by the inherent randomness in model training (K and Søgaard, 2021; Nguyen et al., 2023). This randomness stems from factors such as random initialization, batch sampling, stochastic optimization methods (e.g., dropout (Srivastava et al., 2014)), and even the limitations of floating-point precision. Such randomness can inject significant noise into the influence scores, potentially overwhelming the true signal and rendering the attributions less informative.

One common strategy to address this challenge is to use an *ensemble* approach (Dietterich, 2000), which smooths out noise and improves the signal-to-noise ratio by averaging influence scores across multiple models trained on the same dataset. Early work in this area involved independently trained models (K and Søgaard, 2021; Park et al., 2023;

[3]Beyond the research discussed here, IF's have been extended to analyze the impact of a subset or group of samples (Koh et al., 2019; Hu et al., 2024a), incorporate higher-order information (Giordano et al., 2019a; Basu et al., 2020), and refine influence ranking through normalization techniques (Hammoudeh and Lowd, 2022; Barshan et al., 2020).

Bae et al., 2024), while more recent studies have demonstrated that ensembling different versions of a single model, obtained through variations in dropout, fine-tuning procedures, or checkpoint selection, can further improve attribution quality (Deng et al., 2024a; Wu et al., 2024b). On the other hand, while ensembling is a convenient, plug-and-play technique, it also incurs additional computational cost.

Non-convergence gap. In addition to the issues discussed above, another significant concern is that neural networks may not be trained to full convergence, either intentionally (since early stopping is widely recognized to enhance generalization (Prechelt, 2002; Goodfellow, 2016)) or due to computational constraints. In this context, Bae et al. (2022) systematically analyzes this issue along with the previous ones and argue that the practical implementation of `IF` in deep neural networks does not approximate LOO but instead tracks a different measure, which they term the proximal Bregman response function (PBRF). This quantity can be viewed as a prediction-constrained version of LOO. Notably, while their work does not provide a solution to bridge the non-convergence gap, it offers insight into what `IF` is tracking and contends that this measure can still be valuable for data attribution. More recently, Wu et al. (2024b) proposed Debias and Denoise Attribution (DDA), which addresses the challenge of non-convergence in fine-tuning LLMs. Specifically, DDA improves influence function estimates through a debiasing strategy that removes bias introduced by the pretrained model and a denoising strategy that smooths variations in influence scores across multiple checkpoints.

2.2.2 Improving Scalability

Aside from non-convexity and non-convergence, the computational challenges associated with `IF` present significant bottlenecks in large-scale machine learning applications. In this section, we outline the primary hurdles in scaling `IF` and review techniques to address them.

Recall that computing `IF` involves the following inverse-Hessian-Vector Product (iHVP):

$$\underbrace{H_{\hat{\theta}}^{-1} v}_{O(p^3)} \quad \text{where} \quad H_{\hat{\theta}} = \underbrace{\frac{1}{n} \sum_{i=1}^{n} \nabla_\theta^2 \ell(z_i, \hat{\theta})}_{O(np^2)} \quad \text{and} \quad v = \underbrace{\nabla_\theta \ell(z_{\text{test}}, \hat{\theta})}_{O(p)}.$$

The standard approach of computing a single iHVP (i.e., for a single v) requires calculating the empirical Hessian and then inverting it. This results in a time complexity of $O(np^2 + p^3)$ and a space complexity of $O(p^2)$, which are generally prohibitive even for small neural networks. Moreover, once the iHVP is obtained, the default strategy for identifying influential samples involves scanning the entire training set and computing the per-sample gradient, then taking the dot-product with the iHVP as in Equation (3) to obtain the influence for each sample. This increases the time complexity by $O(np)$.

To reduce computational costs, existing research focuses primarily on three directions: 1) restricting the set of model parameters considered (reducing p); 2) subsampling the training dataset (reducing n); and 3) efficiently approximating the iHVP instead of computing it directly using the standard approach.[4] Among these, the third direction has attracted substantially more research attention. We note that the techniques from these directions are *orthogonal* and can often be combined.

Restricting model parameters. A common approach to scaling up influence functions is to restrict the computation to a subset of model parameters. For instance, several works compute influences only on the last layer (Koh and Liang, 2017; Yeh et al., 2018; Guo et al., 2021). However, Basu et al. (2021) shows that influence functions computed on a single layer may not fully capture the overall influence of training examples.

In the context of LLMs, Grosse et al. (2023) compute influences only for MLP layers while excluding others such as the attention layers. Another promising strategy is to leverage parameter-efficient fine-tuning paradigms common in LLMs, such as Low-Rank Adaptation (LoRA) (Hu et al., 2022a). Performing attribution solely on the LoRA layers substantially reduces the number of parameters under consideration, an approach that has gained increasing popularity in recent studies (Kwon et al., 2024; Zhou et al., 2024).

Filtering training samples. Another strategy for scaling up influence functions is to pre-filter the training dataset to identify a subset of samples that are likely to be influential, thereby avoiding the expensive computation of influence scores for every training example. Guo et al. (2021) reduce the search space to the top-k nearest neighbors of the test point in the embedding space. Grosse et al. (2023) instead leverage TF-IDF (Ramos et al., 2003), a token-matching

[4] Methods in this category also involve (more advanced) strategies of reducing p.

technique, to select sequences most related to a given query, which is conceptually similar to the kNN approach of Guo et al. (2021). However, Grosse et al. (2023) also note that token-matching methods may fail to capture the higher-level abstractions inherent in language models. Other lines of work involve attributing at the level of groups of training points (Ley et al., 2024), and can wrap existing IF methods, reporting speedups based on group granularity.

Efficiently approximating iHVP. The two directions above do not tackle the fundamental challenge of computing the iHVP, which is the primary bottleneck in scaling `IF`'s. Here, we discuss three approaches to efficiently approximate the iHVP: 1) using compute-friendly approximations of the Hessian; 2) employing iterative algorithms; and 3) applying gradient projection methods.

To derive the time and space complexities of different methods, we consider the model to be a multilayer perceptron (MLP) with L layers, each having a weight matrix $W_l \in \mathbb{R}^{a\times a}$ of identical shape across layers. The number of parameters per layer is $d = a^2$. At layer l, we denote $s_l \in \mathbb{R}^a$ as the pre-activation, $a_l \in \mathbb{R}^a$ as the activation, and $\theta_l = \text{vec}(W_l)$ as the flattened weights. We further define the shorthand $g_{s_l} = \nabla_{s_l}\ell(\hat{\theta}; z)$ and $g_{\theta_l} = \nabla_{\theta_l}\ell(\hat{\theta}; z)$ to indicate loss gradients w.r.t. s_l and θ_ℓ. We continue to use n to denote the total number of training points.

Compute-friendly approximations of the Hessian. A common strategy to reduce the computational cost of iHVP is to approximate the Hessian with a more tractable matrix. As discussed in "Singularity of Hessian," one popular alternative is the Fisher Information Matrix (FIM), which approximates the second-order Hessian with first-order gradient information only. Another widely adopted technique is to use a layer-wise block-diagonal approximation of the FIM, ignoring interactions between parameters across different layers. Concretely, the full Hessian is approximated by $\text{diag}\{G_1, \cdots, G_L\}$, where

$$G_l = \mathbb{E}_z[\underbrace{\nabla_{\theta_l}\ell(\hat{\theta}; z)}_{O(d)} \nabla_{\theta_l}\ell(\hat{\theta}; z)^\top] = \underbrace{\mathbb{E}_z[g_{\theta_l} g_{\theta_l}^\top]}_{O(nd^2)}.$$

We refer to this approach as *FIM + block-diagonal approximation*, or BLOCK DIAGONAL for short. This approach alone is capable of cutting asymptotic iHVP runtime and storage notably. Specifically, computing and inverting all G_ℓ results in an overall time complexity of $O(nLd^2 + Ld^3)$, as opposed to $O(nL^2d^2 + L^3d^3)$ when all $p = Ldp = Ld$ parameters are considered. Meanwhile, the Hessian and its inverse would typically require $O(p^2) = O(L^2d^2)$ in space, whereas computing these layer-wise requires a space complexity of $O(Ld^2)$ instead.[5]

We further discuss a few advanced extensions of this approach in several recent works.

- K-FAC (Martens and Grosse, 2015), or Kronecker-Factored Approximate Curvature, is an efficient approximation to the aforementioned FIM + block-diagonal. The key idea behind K-FAC is that the parameter gradients within each layer of an MLP can be further factored into $g_{\theta_l} = a_{l-1} \otimes g_{s_l}$, where $\otimes$ denotes the Kronecker product:

$$\begin{aligned} G_l &= \mathbb{E}_z[(a_{l-1} \otimes g_{s_l})(a_{l-1} \otimes g_{s_l})^\top] = \mathbb{E}_z[(a_{l-1}a_{l-1}^\top) \otimes (g_{s_l}g_{s_l}^\top)] \\ &\approx \underbrace{\mathbb{E}_z[a_{l-1}a_{l-1}^\top]}_{O(na^2)} \otimes \underbrace{\mathbb{E}_z[g_{s_l}g_{s_l}^\top]}_{O(na^2)} = A_{l-1} \otimes S_l. \end{aligned}$$

  This final step assumes approximate independence between the activations a_{l-1} and the backpropagated gradients g_{s_l}. While this is not strictly true, Martens and Grosse (2015) justify this simplification using heuristic arguments based on graphical models, supported empirically— and it is theoretically exact in simpler linear-Gaussian regimes (Bernacchia et al., 2018). Since the inverse of the block-diagonal matrix is also block-diagonal, the $G^{-1}v$ approximation can be performed layer-wise, and the separation of A_{l-1} and S_l enables the following useful application of identities, where $V_l \in \mathbb{R}^{a\times a}$ and $v_l = \text{vec}(V_l)$:

$$G_l^{-1}v_l = (A_{l-1} \otimes S_l)^{-1}v_l = (\underbrace{A_{l-1}^{-1}}_{O(a^3)} \otimes \underbrace{S_l^{-1}}_{O(a^3)})v_l = \text{vec}(\underbrace{S_l^{-1}V_lA_{l-1}^{-1}}_{O(a^3)}).$$

  Estimating A_{l-1} and S_l thus requires $O(na^2)$ runtime, while inversion requires $O(a^3)$. The additional matrix multiplications cost $O(a^3)$. Space requirements scale with $O(a^2)$, the size of each matrix.

  Putting this together for L layers, K-FAC enjoys a time complexity of $O(nLd+Ld^{3/2})$, owing to the Kronecker factorization of each layer's parameters into input and output dimensions, as well as a space complexity of $O(Ld) = O(p)$, which is the size of the model.

[5]We assume that in practice $L \ll d$, and $O(Ld)$ storage requirements to compute gradients thus vanish. Similarly, computing G_l technically requires $O(nLd + nd^2) = O(nd^2)$ time when performed layer-wise.

- EK-FAC (Grosse et al., 2023), the eigenvalue-corrected K-FAC approximation originally proposed by George et al. (2018), eigendecomposes A_{l-1} and S_l and efficiently computes the iHVP by inverting the resulting diagonal form. Asymptotic time and space complexities remain the same as vanilla K-FAC, though the resulting approximation is provably better. EK-FAC also provides a way to handle damped iHVPs in the form of $(G_l + \lambda_l I)^{-1}$, which is equivalent to adding λ_l to each of the eigenvalues.
- DATAINF (Kwon et al., 2024) proposes to swap the order of operations in the expression for $(G_l + \lambda_l I)^{-1}$, from the standard inverse of the FIM expectation, to an expectation over per-sample FIM inverses, via

$$\left(\frac{1}{n}\sum_{i=1}^{n} g_{i,\theta_l}\, g_{i,\theta_l}^\top + \lambda_l I\right)^{-1} \approx \frac{1}{n}\sum_{i=1}^{n}\Big(g_{i,\theta_l}\, g_{i,\theta_l}^\top + \lambda_l I\Big)^{-1},$$

  where we extend our shorthand to $g_{i,\theta_l} = \nabla_{\theta_l}\ell(\theta, z_i)$. While a theoretical justification for this rearrangement is yet to be established, Kwon et al. (2024) derive error bounds under mild assumptions (bounded gradients and positive damping), showing that the approximation error decreases with stronger damping and is controlled by the layer's parameter dimension. The main practical benefit is that it enables direct use of the Sherman–Morrison formula, yielding a much more efficient closed form expression for $(G_l + \lambda_l I)^{-1}$ of

$$\frac{1}{n}\sum_{i=1}^{n}\big(g_{i,\theta_l}\, g_{i,\theta_l}^\top + \lambda_l I\big)^{-1} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\lambda_l}\left(I - \frac{g_{i,\theta_l} g_{i,\theta_l}^\top}{\lambda_l + g_{i,\theta_l}^\top g_{i,\theta_l}}\right).$$

  The full influence estimate $-v_l^\top (G_l + \lambda_l I)^{-1} g_{k,\theta_l}$ for the l^{th}-layer reduces the expression to[6]

$$\frac{1}{n\lambda_l}\sum_{i=1}^{n}\left(\underbrace{\frac{(v_l^\top g_{i,\theta_l})(g_{i,\theta_l}^\top g_{k,\theta_l})}{\lambda_l + g_{i,\theta_l}^\top g_{i,\theta_l}}}_{O(d)} - \underbrace{v_l^\top g_{k,\theta_l}}_{O(d)}\right),$$

  which now consists only of inner-products between layer-wise gradients in $\mathbb{R}^d$ space, reducing runtime complexity to $O(nd)$ per layer. For an estimate of the influence of a single training point on a single test point, DataInf has an overall time complexity of $O(nLd)$ and space complexity of $O(Ld)$, i.e., for each gradient term in the expression.

Iterative algorithms. Another major source of computational cost is the inversion of the Hessian (or its approximation). Even with block-diagonal approximations, direct inversion can be expensive. Iterative algorithms have been developed to compute the iHVP more efficiently.

- LISSA: To approximate $H_{\hat{\theta}}^{-1} v$, Koh and Liang (2017) employs LiSSA, an algorithm for stochastic optimization proposed in Agarwal et al. (2017)). Given a set of training batches $z_{s_1}, \ldots, z_{s_t}$ sampled uniformly at random, the method iterates

$$\tilde{H}_0^{-1} v = v; \quad \tilde{H}_j^{-1} v = v + \Big(I - \nabla_\theta^2 \ell(z_{s_j}, \hat{\theta})\Big)\tilde{H}_{j-1}^{-1} v,$$

  for $j = 1, \ldots, t$. This procedure is repeated r times, and the results are averaged to yield the final iHVP approximation. The efficiency of LiSSA partly arises from the fact that the Hessian-vector product $\nabla_\theta^2 \ell(z, \hat{\theta}) v$ can be computed in $O(p)$ time (Pearlmutter, 1994), for $O(rt)$ samples. Practically, Koh and Liang (2017) note that $rt \approx n$ in their settings gives accurate results, resulting in an overall online time complexity of $O(np)$ and space complexity of $O(p)$. Subsequent works have reduced the effective number of repetitions r by parallelizing the computation (Guo et al., 2021) or using advanced sampling techniques (e.g., centroid-based sampling via K-means) to improve the robustness of the Hessian approximation (Koh et al., 2024).
- SCHULZ: Zhou et al. (2024) adopt Schulz's iteration (Herzberger and Petković, 1990) to compute the inverse of the Hessian approximation G. The (per-layer) iterative update is given by

$$\tilde{G}_0^{-1} \approx G^{-1}; \quad \tilde{G}_j^{-1} = \underbrace{\tilde{G}_{j-1}^{-1}\Big(2I - G\,\tilde{G}_{j-1}^{-1}\Big)}_{O(d^3)}.$$

  Although it does not improve the asymptotic complexity of matrix inversion, this method comes with numerical stability guarantees and is relatively insensitive to the initial guess.[7] Computing G requires $O(nd^2)$ time per

[6] N.B. the original DataInf notation has been heavily adapted to match the format presented in this section.

[7] Initializing with the identity or a random gaussian matrix resulted in convergence within $t < 20$ steps according to Zhou et al. (2024).

layer, and further applying t iterative updates as shown above at $O(d^3)$ time across L layers results in a time complexity of $O(nLd^2 + tLd^3)$ and a space complexity of $O(d^2)$.

Gradient projection. A third approach to reduce iHVP cost is to project the gradients onto a lower-dimensional subspace, thereby reducing the size of the matrices involved in the computation. We denote $\tilde{p}$ and $\tilde{d}$ as the projected number of parameters, a fraction of the original number of parameters p and d. We do likewise for projected activations $\tilde{s}_l$ and $\tilde{a}_{l-1}$. Two common techniques are:

- RANDOM (Schioppa et al., 2022; Park et al., 2023): In this approach, a projection matrix $P \in \mathbb{R}^{\tilde{p} \times p}$ is drawn from a suitable distribution (e.g., Gaussian), where $\tilde{p} \ll p$ represents the reduced dimensionality. Replacing gradients g_θ with projected gradients Pg_θ, the Hessian is approximated by:

$$\tilde{G} = \frac{1}{n} \sum_{i=1}^{n} \underbrace{(Pg_\theta)(Pg_\theta)^\top}_{O(p\tilde{p}+\tilde{p}^2)},$$

  which rearranges to $\tilde{G} = PGP^\top$, and can be interpreted as a restriction of the Hessian to a subspace spanned by the rows of P. Likewise, train and test gradients are projected onto this space in $O(p\tilde{p})$ time and space to compute iHVPs and influence scores.

  The cost of computing and inverting $\tilde{G}$ is $O(np\tilde{p}+\tilde{p}^3)$ which simplifies to $O(np\tilde{p})$ assuming $\tilde{p} \ll p, n$. Should this be performed layerwise with the block diagonal approximation, we have $O(nLd + nLd\tilde{d})$ runtime, as gradients must still be computed for the entire model using $O(Ld)$ time. Space scales with $O(Ld + d\tilde{d})$ owing to an initial full gradient computation followed by projection occurring at the layer level.

- ARNOLDI: While the Johnson-Lindenstrauss lemma (William and Lindenstrauss, 1984) ensures that the inner products between gradients are approximately preserved under random i.i.d. projection, if the projection subspace is not invariant under G (i.e., G applied to vectors in this subspace generates significant components orthogonal to the subspace), the projected Hessian $PGP^\top$ implicitly neglects these orthogonal components, which leads directly to approximation errors. This motivates Arnoldi iteration. Schioppa et al. (2022) propose to use Arnoldi iteration (Arnoldi, 1951) to approximate the top-$\tilde{p}$ (in magnitude) eigenvalues of $H_{\hat{\theta}}$, and use the corresponding eigenvectors as the rows of the projection matrix $P \in \mathbb{R}^{\tilde{p} \times p}$. Since this subspace is spanned by the eigenvectors of $H_{\hat{\theta}}$, it is by definition $H_{\hat{\theta}}$-invariant, i.e., vectors will remain in the subspace when transformed by $H_{\hat{\theta}}$. Another highly desirable property of top-$\tilde{p}$ eigenvector projection is that $PH_{\hat{\theta}}P^\top$ simplifies to the diagonal matrix of eigenvalues, Λ, which is trivially invertible in $O(\tilde{p})$ time. The iHVP then becomes

$$(PH_{\hat{\theta}}P^\top)^{-1}(Pv_l) = \underbrace{\Lambda^{-1}}_{O(\tilde{p})} \underbrace{Pv_l}_{O(p\tilde{p})}.$$

  While projecting onto the top-$\tilde{p}$ eigenvectors of the Hessian matrix is highly desirable[8], this step comes with extra computational cost, as we will now outline. The Arnoldi iteration starts by iteratively constructing the following $\tilde{p}^{\text{th}}$ order Krylov subspace[9] of the Hessian:

$$\text{Span}\{v, H_{\hat{\theta}}v, \underbrace{H_{\hat{\theta}}^2 v}_{O(Bp)}, \ldots, H_{\hat{\theta}}^{\tilde{p}} v\},$$

  where $v \in \mathbb{R}^p$ is randomly drawn. Intuitively, the sequence above naturally converges to the largest eigenvector of $H_{\hat{\theta}}$, and for sufficiently large $\tilde{p}$, we may also expect an orthonormal basis of the subspace to span good approximations of the top-$\tilde{p}$ eigenvectors of $H_{\hat{\theta}}$. While iterating on the above, Arnoldi orthogonalizes each new term w.r.t. each of the previous terms, and subsequently normalizes, to form said orthonormal basis. It therefore uses $\tilde{p}$ iterations, each of which requires an HVP and its orthogonalization with respect to every existing basis vector, costing $O(Bp\tilde{p} + p\tilde{p}^2)$ runtime. Note that the HVP is not computed over the full training data, nor individual training samples as in LiSSA, but rather over batches of size B. Concurrently, a matrix A is computed containing information about inner products between each new HVP and the basis vectors up until that iteration. Schioppa et al. (2022) then eigendecomposes A and projects the eigenvectors onto the orthonormal basis to approximate the top-$\tilde{p}$ eigenvectors of $H_{\hat{\theta}}$. Eigendecomposing A entails $O(\tilde{p}^3)$ runtime,

[8] Theoretically, the top-$\tilde{p}$ eigenpairs also yield the best rank-$\tilde{p}$ approximation of the full (symmetric) Hessian: specifically, $P^\top \tilde{H} P$ is closest in both the Frobenius and spectral norms to $H_{\hat{\theta}}$, when $P \in \mathbb{R}^{\tilde{p} \times p}$ consists of the top-$\tilde{p}$ eigenvectors, and $\tilde{H} \in \mathbb{R}^{\tilde{p} \times \tilde{p}}$ is the diagonal matrix of corresponding eigenvalues (Eckart and Young, 1936).

[9] Technically speaking, Schioppa et al. (2022) constructs a $> \tilde{p}^{\text{th}}$ order subspace first, before selecting the top-$\tilde{p}$ eigenpairs.

while projecting gradients with P requires $O(p\tilde{p})$, meaning the total runtime remains at $O(Bp\tilde{p} + p\tilde{p}^2)$, or equivalently $O(BLd\tilde{d} + Ld\tilde{d}^2)$ in the block-diagonal setting. Space scales with $O(p\tilde{p} + \tilde{p}^2)$, or $O(Ld\tilde{d} + L\tilde{d}^2)$.

- LOGRA (Choe et al., 2024): Low-Rank Gradient Projection improves space and time complexity of gradient projection by separating layer-wise projections into two smaller projections at the input and output of the layer (similarly exploited in K-FAC). In doing so, the full projected gradient may be rewritten as the following, where P_i and P_o represent the projections applied to the input activations and output pre-activations, respectively:

$$Pg_{\theta_l} = (P_i \otimes P_o)(a_{l-1} \otimes s_l) = P_i a_{l-1} \otimes P_o s_l = \tilde{a}_{l-1} \otimes \tilde{s}_l,$$

or the Kronecker product of projected activations/gradients in the layer, where $\tilde{a}_{l-1}, \tilde{s}_l \in \mathbb{R}^{\tilde{a}}$ represent projected activations. As for the Hessian estimate $\tilde{G}_l = PG_lP^\top$, applying the same identities used to derive K-FAC yields:

$$\begin{aligned}\tilde{G}_l &= \mathbb{E}_z[(Pg_{\theta_l})(Pg_{\theta_l})^\top] = \mathbb{E}_z[(\tilde{a}_{l-1} \otimes \tilde{s}_l)(\tilde{a}_{l-1} \otimes \tilde{s}_l)^\top] \\ &\approx \mathbb{E}_z[\tilde{a}_{l-1}\tilde{a}_{l-1}^\top] \otimes \mathbb{E}_z[\tilde{s}_l\tilde{s}_l^\top] = \tilde{A}_{l-1} \otimes \tilde{S}_l,\end{aligned}$$

with forward and backward uncentered covariance matrices $\tilde{A}_{l-1}, S_l \in \mathbb{R}^{\tilde{a}\times\tilde{a}}$. Overall, the derivations in this section are directly analogous to those in K-FAC, with the final iHVP form of $\tilde{G}_l^{-1}\tilde{v}_l = \text{vec}(\tilde{S}_l^{-1}\tilde{V}_l\tilde{A}_{l-1}^{-1})$, where $\tilde{v}_l = Pv_l$ is the projected test gradient w.r.t. the layer's parameters. We denote the matrix form (taking input and output shapes into account) of $\tilde{v}_l$ as $\tilde{V}_l$. Projecting onto each of these spaces *separately* requires $O(\sqrt{d\tilde{d}})$ time, in contrast to standard projection of $g_{\theta_l} \in \mathbb{R}^d$ to $\tilde{g}_{\theta_l} \in \mathbb{R}^{\tilde{d}}$ which requires $O(d\tilde{d})$. The full runtime complexity of LoGra is $O(nLd/B + nL\sqrt{d\tilde{d}} + L\sqrt{\tilde{d}}^3)$, owing to per-sample gradient computation (batch size B), the computation of $\tilde{A}_{l-1}$ and $\tilde{S}_l$ followed by inversion of $\tilde{G}_l$ for each layer. Assuming $\sqrt{\tilde{d}} \ll n$, the first term dominates, giving $O(nL\sqrt{d\tilde{d}})$ runtime for iHVP pre-computation. As for space complexity, we still require $O(Ld)$ to store the model, which dominates the smaller space requirements of $O(\sqrt{d\tilde{d}})$ for the projection matrices and $O(\tilde{d})$ for matrix inversions. If projected training gradients are cached for reuse, an additional disk storage cost of $O(nL\sqrt{\tilde{d}})$ is incurred. In summary, LoGra uses a combination of gradient projection, block-diagonal approximation, and K-FAC approximation.

2.2.3 Comparison of IHVP Computational Complexities

To clarify comparisons across methods, Table 3 presents complexities for computing the influence of a *single training point* on a *single test point*, v. We distinguish clearly between **pre-compute** complexity, to represent any Hessian related computational cost (such as IHVPs, $H_{\hat{\theta}}^{-1}v$, or inverse Hessians, $H_{\hat{\theta}}^{-1}$), and **per-vector (online)** complexity, the cost of attributing to each training point, given access to the pre-computed data relating to the specific test point. We assume that the methods utilize the FIM + block-diagonal approximation of the Hessian wherever possible (i.e., besides the standard LiSSA method). Note that in many cases, parallelization can exchange runtime complexity for space complexity $O(nLd)$ for increased space complexity $O(nLd)$, beneficial in large-scale settings, i.e. L can be transferred from runtime to space complexity if parallelized, or online runtime complexity $O(nLd)$ can be interchanged with increased space complexity.

- EK-FAC, SCHULZ, and BLOCK DIAGONAL exploit structured approximations, significantly reducing complexities from exact methods $O(nL^2d^2 + L^3d^3)$ to $O(nLd + Ld^{3/2})$, $O(nLd^2 + tLd^3)$, and $O(nLd^2 + Ld^3)$, respectively. Assuming terms with n dominate in practical settings, the $O(nL^2d^2)$ pre-compute complexity for exact influence is reduced to $O(nLd^2)$ with BLOCK DIAGONAL and SCHULZ, and further reduced to $O(nLd)$ with EK-FAC.
- DATAINF shifts the IHVP computation primarily online. However, it can be structured to transfer complexity offline, reducing the online complexity to $O(Ld)$ while incurring $O(nLd)$ offline costs by factoring out gradient terms, denoted with DATAINF (W/PRE) in Table 3. This is necessary if attributing to multiple training points, allowing online time complexity to scale with $O(nLd)$ instead of $O(n^2Ld)$, as we further describe below.
- LISSA typically moves computation online with complexity $O(rtLd)$. However, pre-computation is possible by caching intermediate vectors ($s_{\text{test}} = H_{\hat{\theta}}^{-1}v_{\text{test}}$), allowing the online runtime to be reduced to $O(Ld)$ per training point, or $O(nLd)$ across multiple training points. To compute importance for *multiple test points*, LISSA's pre-compute complexity grows with m, the number of test points, to $O(mrtLd)$, as does online runtime, which scales linearly with m.

- Projection methods (RANDOM, ARNOLDI, LOGRA) further minimize pre-compute complexity, with slight increases in per-vector complexity, making them indispenable in applications where full gradient computations are prohibitive (e.g., LLMs). Of these, LoGra offers the most extensive speedups, combining principles from EK-FAC (factorization of layerwise gradients into forward and backward passes) with gradient projection to achieve $O\left(nL\sqrt{d\tilde{d}}\right)$ pre-compute complexity.

Attributing to multiple training points. With $p = Ld$ parameters, attributing to *multiple training points* increases online compute from $O(Ld)$ to $O(nLd)$ for all methods (this may be transferred to space complexity when utilizing parallelization). Online cost may further increase to $O(mnp)$ if scores are required across m test points.

Pre-compute costs for multiple test points. While computing the influence of multiple training points increase online runtime, computing these training data influences on multiple test points has varying effects on the pre-compute complexities across the methods. In general, methods that compute IHVPs directly typically scale poorly to multiple test points, as they save little re-usable information such as the result of $H_{\hat{\theta}}^{-1}$.

- EXACT influence adds $O(mp^2)$ to compute $H_{\hat{\theta}}^{-1}v$ for all m test points, since $H_{\hat{\theta}}^{-1}$ is explicitly computed. This new term remains dominated by the original $O(p^3)$ terms, and complexities are unchanged. In layer-wise settings, this increase is $O(mLd^2)$, which vanishes for BLOCK DIAGONAL and SCHULZ, where the pre-compute time complexities include larger $O(nLd^2)$ terms.
- EK-FAC requires $O(Ld^{3/2})$ pre-compute runtime for each new IHVP requiring. Thus, the precompute complexity with m IHVP operations is increased to $O(nLd + mLd^{3/2})$, which remains unchanged if $m\sqrt{d} \ll n$.
- DATAINF allows for factorization of $v_\ell^\top$ and g_{k,θ_ℓ} terms. As such, the remaining expression comprising n training gradients can first be computed in $O(nLd)$ time. Pre-computing all IHVPs comes with an additional $O(mLd)$ pre-compute cost, which again vanishes.
- LISSA solves each particular IHVP directly, in $O(rtLd)$ time, which scales poorly to $O(mrtLd)$ for multiple test points, since, as noted above, there is little intermediate, re-usable information across test points.
- Projection methods (RANDOM, ARNOLDI, LOGRA) explicitly compute the (projected) Hessian inverse and thus incur negligible additional pre-compute costs for multiple IHVPs (as n dominates m).

Recommendations. Traditional influence function research has primarily focused on addressing the computational bottleneck of Hessian inverse computation, which represents the main challenge for training runs with small-scale datasets and many training epochs. However, modern large-scale deep learning, especially for large language models,

Table 3: Comparison of time and space complexity of different methods to compute the influence of a *single training point* on a *single test point*. The model is an MLP with L layers of identical shape, each containing d parameters (or $\tilde{d}$ projected parameters).

Category	Method	Pre-compute Time	Pre-compute Space	Time	Space
Exact	NAIVE	$O(nL^2d^2 + L^3d^3)$	$O(L^2d^2)$	$O(Ld)$	$O(Ld)$
Hessian Approx.	BLOCK DIAGONAL	$O(nLd^2 + Ld^3)$	$O(d^2)$	$O(Ld)$	$O(Ld)$
	EK-FAC	$O(nLd + Ld^{3/2})$	$O(Ld)$	$O(Ld)$	$O(Ld)$
	DATAINF	-	-	$O(nLd)$	$O(Ld)$
	DATAINF (W/PRE)	$O(nLd)$	$O(Ld)$	$O(Ld)$	$O(Ld)$
Iterative	LISSA/FASTIF	-	-	$O(rtLd)$	$O(Ld)$
	SCHULZ	$O(nLd^2 + tLd^3)$	$O(d^2)$	$O(Ld)$	$O(Ld)$
Projection	RANDOM	$O(nLd\tilde{d})$	$O(Ld\tilde{d})$	$O(Ld\tilde{d})$	$O(Ld\tilde{d})$
	RANDOM (W/PRE)	$O(nLd\tilde{d})$	$O(nL\tilde{d} + Ld\tilde{d})$	$O(L\tilde{d})$	$O(L\tilde{d})$
	ARNOLDI	$O(BLd\tilde{d} + Ld\tilde{d}^2)$	$O(Ld\tilde{d} + L\tilde{d}^2)$	$O(Ld\tilde{d})$	$O(Ld\tilde{d})$
	ARNOLDI (W/PRE)	$O(BLd\tilde{d} + Ld\tilde{d}^2 + nLd\tilde{d})$	$O(nL\tilde{d} + Ld\tilde{d} + L\tilde{d}^2)$	$O(L\tilde{d})$	$O(L\tilde{d})$
	LOGRA	$O(nL\sqrt{d\tilde{d}})$	$O(Ld)$	$O(Ld + \sqrt{d\tilde{d}})$	$O(Ld)$
	LOGRA (W/PRE)	$O(nL\sqrt{d\tilde{d}})$	$O(nL\sqrt{\tilde{d}} + Ld)$	$O(L\sqrt{\tilde{d}})$	$O(Ld)$

presents additional challenges. Simply improving Hessian inverse efficiency is insufficient; the critical bottleneck has extended to computing dot-products between per-sample gradients and the iHVP, which requires per-sample gradient information. The gradient decomposition techniques from K-FAC literature can significantly improve the computational efficiency of this process. For contemporary ML applications requiring real-time influence estimation across numerous inference queries, we recommend a precomputation-storage paradigm: computing and storing iHVPs for each training sample to disk, then performing efficient dot-products with test gradients at inference time. However, this approach demands substantial storage capacity proportional to the training set size and model dimensionality. To mitigate storage requirements while preserving influence estimation quality, projection methods can be used to compress iHVPs into lower-dimensional representations. This storage-projection framework is so far the most practical approach for scalable influence computation in production environments, trading one-time preprocessing costs and storage overhead for dramatically reduced inference-time latency. Moving forward, there are promising opportunities to develop more effective projection techniques that maximize influence estimation accuracy under constrained storage budgets.

2.2.4 TRAK: A Variant of Influence Function

Recently, TRAK (Park et al., 2023) has emerged as one of the most popular data attribution approaches, finding successful application in a range of generative models due to its effectiveness and scalability. While TRAK is proposed as an efficient estimator of datamodels (Ilyas et al., 2022) (discussed in Section 2.5), it is closer to IF on a methodology level. In particular, we begin by reviewing the concept of the *one-step Newton update*—a fundamental idea underlying TRAK that has long been recognized for its close connection to IF (Pregibon, 1981; Wojnowicz et al., 2016; Koh et al., 2019). Next, we discuss the key design choices behind TRAK. Finally, we examine its applications across various generative models.

One-step Newton update. At the core of TRAK is the one-step Newton update, detailed below. Let $\mathcal{L}_{-z}$ denote the loss function computed without the point z, and let H_{-z} be its corresponding Hessian. Starting from the full-data estimate $\hat{\theta}$, the Newton-Raphson method (Ypma, 1995) updates the parameters as

$$\theta_{\text{NS}} \coloneqq \hat{\theta} - H_{-z,\hat{\theta}}^{-1} \nabla_\theta \mathcal{L}_{-z}(\hat{\theta}), \tag{4}$$

where, by the first-order optimality condition $\nabla_\theta \mathcal{L}(\hat{\theta}) = 0$, we have

$$\nabla_\theta \mathcal{L}_{-z}(\hat{\theta}) = -\frac{1}{n} \nabla_\theta \ell(z, \hat{\theta}).$$

Comparing Equation (4) with Equation (2), we observe that the IF and one-step Newton update serve as approximations of one another, assuming that $H_{-z,\hat{\theta}}^{-1} \approx H_{\hat{\theta}}^{-1}$.

Design choices of TRAK. We first consider the original formulation of TRAK based on the one-step Newton update in Equation (4), and then describe the key design choices that enable efficient and scalable attribution for large-scale models.

For clarity, we derive TRAK in the context of a multi-class classification problem, which also forms the basis for its application to language models. Here, the loss function ℓ is the cross-entropy loss, and $\mathcal{L}$ represents the *sum* (rather than the average) of the loss over the training set D. Let p_i denote the predicted probability of the correct class for sample z_i, and consider the target function

$$f(z_{\text{test}}; \theta) = \log \frac{p}{1-p},$$

referred to as the *multi-class margin* function (Engstrom et al., 2024) for a test point z_{test}. This function is particularly useful because the cross-entropy loss can be succinctly expressed as $\ell(y_i, f(z_i; \theta)) = \log(1 + e^{-f(z_i;\theta)})$. We also denote $g_i = \nabla_\theta f(z_i; \hat{\theta})$ and $g_{\text{test}} = \nabla_\theta f(z_{\text{test}}; \hat{\theta})$. It is straightforward to compute that

$$\begin{aligned}
\nabla_\theta \ell(z_i, \hat{\theta}) &= \frac{\partial \ell(y_i, f(z_i; \hat{\theta}))}{\partial f(z_i; \hat{\theta})} g_i = (p_i - 1) g_i, \\
H_{-z_i,\hat{\theta}} &= \sum_{j \neq i} \left(\frac{\partial \ell(y_j, f(z_j; \hat{\theta}))}{\partial f(z_j; \hat{\theta})} \nabla_\theta^2 f(z_j; \hat{\theta}) + \frac{\partial^2 \ell(y_j, f(z_j; \hat{\theta}))}{\partial f(z_j; \hat{\theta})^2} g_j g_j^\top \right) \\
&= \sum_{j \neq i} \left(\frac{\partial \ell(y_j, f(z_j; \hat{\theta}))}{\partial f(z_j; \hat{\theta})} \nabla_\theta^2 f(z_j; \hat{\theta}) + p_j(1 - p_j) g_j g_j^\top \right).
\end{aligned}$$

With these quantities, the *ideal* TRAK score of sample z_i for a test sample z_{test} is

$$\mathcal{I}^{\text{ideal}}_{\texttt{TRAK}}(z_{\text{test}})_i = g_{\text{test}}^\top \left(\sum_{j \neq i} \frac{\partial \ell(y_j, f(z_j;\hat{\theta}))}{\partial f(z_j;\hat{\theta})} \nabla^2_\theta f(z_j;\hat{\theta}) + p_j(1-p_j) g_j g_j^\top \right)^{-1} g_i(p_i - 1). \tag{5}$$

We now discuss the key design choices adopted by Park et al. (2023).

Linearize the target function. The first step is to linearize the target function around the optimal model parameter $\hat{\theta}$. Formally, TRAK uses the approximation

$$f(z_i;\theta) \approx f(z_i;\hat{\theta}) + \nabla_\theta f(z_i;\hat{\theta})^\top (\theta - \hat{\theta}).$$

This linearization effectively eliminates the potentially non-PSD component of the Hessian in Equation (5), since $\nabla^2_\theta f(z_j;\hat{\theta}) = 0$. This step is equivalent to the FIM approximation discussed in Sections 2.2.1 and 2.2.2. Using the Sherman-Morrison formula (see Hu et al. (2024a) for details), the TRAK score simplifies to

$$\frac{g_{\text{test}}^\top (G^\top R G)^{-1} g_i (1-p_i)}{1 - h_i},$$

where $G = [g_1, \cdots, g_n]^\top \in \mathbb{R}^{n \times p}$ is the stacked gradients, R is a diagonal matrix with entries $p_i(1-p_i)$, and $h_i = 1 - g_i^\top (G^\top R G)^{-1} g_i \, p_i(1-p_i)$ is the leverage score for sample z_i.

Omit scaling terms. Park et al. (2023) find empirically that both the leverage score in the denominator and the diagonal matrix R have little effect on the final estimates, and thus omit them from the estimator. Ignoring the leverage score essentially reverts the formula to that of IF. The TRAK score thus further simplifies to

$$g_{\text{test}}^\top (G^\top G)^{-1} g_i (p_i - 1).$$

Random projection. To reduce memory and computational costs, TRAK leverages random projection. Rather than storing and computing with the full high-dimensional gradient $g_i \in \mathbb{R}^p$, TRAK computes projected gradients $\phi_i = P^\top g_i$ and $\phi_{\text{test}} = P^\top g_{\text{test}}$, where $P \sim \mathcal{N}(0,1)^{p \times k}$. The score for sample z_i is then given by

$$\phi_{\text{test}}^\top (\Phi^\top \Phi)^{-1} \phi_i (p_i - 1),$$

with $\Phi = [\phi_1, \cdots, \phi_n]^\top \in \mathbb{R}^{n \times k}$ representing the stacked projected gradients. In practice, k is chosen to be on the order of tens of thousands—substantially smaller than the original parameter count—thereby greatly enhancing computational efficiency. In fact, random projection allows TRAK to directly compute the inverse matrix without relying on additional acceleration techniques discussed in Section 2.2.2. Nonetheless, while this method boosts efficiency without significant performance loss empirically, the justification based on the Johnson-Lindenstrauss lemma, as noted by Schioppa et al. (2022), warrants further examination (see "gradient projection" in Section 2.2.2 for discussion).

Ensemble. Finally, TRAK employs an ensemble approach to enhance robustness. By averaging scores from different reference models, either retrained on different subsets or using distinct checkpoints, the method mitigates the inherent randomness of individual model training. Formally, given M reference models $\{\hat{\theta}_m\}_{m=1}^M$ and M corresponding projection matrices $\{P_m \sim \mathcal{N}(0,1)^{p \times k}\}_{m=1}^M$, the *actual* TRAK score for a test sample z_{test} is defined as

$$\mathcal{I}_{\texttt{TRAK}}(z_{\text{test}}) = \left(\frac{1}{M} \sum_{m \in [M]} \left(\phi_{m,\text{test}}^\top \left(\Phi_m^\top \Phi_m \right)^{-1} \Phi_m^\top \right) \right) \cdot \left(\frac{1}{M} \sum_{m \in [M]} Q_m \right), \tag{6}$$

where

$$\phi_{m,i} = P_m^\top \nabla_\theta f(z_i;\hat{\theta}_m), \quad \phi_{m,\text{test}} = P_m^\top \nabla_\theta f(z_{\text{test}};\hat{\theta}_m), \quad \Phi_m = [\phi_{m,1}, \cdots, \phi_{m,n}],$$

and $Q_m = \text{diag}\,(p_{i,m} - 1)$ with $p_{i,m}$ being the predicted probability of the correct class for sample z_i according to $\hat{\theta}_m$. This Q matrix is typically referred to as the *output-to-loss gradient* (Park et al., 2023), defined by

$$Q =: \text{diag}\left(\left\{ \frac{\partial \ell(y_i, f(z_i;\hat{\theta}))}{\partial f(z_i;\hat{\theta})} \right\} \right).$$

Note that the original implementation of TRAK constructs the Q matrix with diagonal entries $1 - p_i$, differing in sign from the actual output-to-loss gradient. This is because the target function here, the multi-class margin function, is maximized (i.e., higher values are better), as opposed to common target functions in IF, such as the test loss, where lower values are preferable. Consequently, Park et al. (2023) consider the negative LOO difference $f(z_{\text{test}};\hat{\theta}) - f(z_{\text{test}};\hat{\theta}_{-z_i})$ so that the interpretation of TRAK scores align with the convention in data attribution: a positive score indicates a helpful sample and a negative score indicates a harmful one. For target functions that we seek to minimize (see Table 5 for examples) , no sign modification is necessary.

Applications of TRAK in generative models. TRAK can be extended to generative tasks by designing a proper target function for different loss functions of various generative tasks.

Language models. The core challenge and research direction to apply TRAK on language models is how to scale to large (e.g., billion-level parameters) models and large datasets while preserving high effectiveness.

DsDm (Engstrom et al., 2024) utilizes the multi-class margin for token predictions, following the approach in TRAK, to attribute examples. To scale to large settings, the paper adopts an approach to use a small proxy model to compute the data scores and then apply them to downstream tasks on larger models. DsDm (Engstrom et al., 2024) utilizes TRAK as an estimator for linear datamodels on data selection tasks. Specifically, DsDm calculates individual data scores using a small proxy model, i.e., a pre-trained 125M parameter GPT-2 style language model on 6 billion tokens, and selects the top-K positively influential samples for training larger models with a maximum budget of 1.3B parameters.

Another study, TrackStar, combines several techniques to refine the computation of the gradient-based influence score and scales the computation to large setups of 8B pertaining LLMs. Specifically, TrackStar defines the influence score as

$$\mathcal{I}(z, z_q) = \bar{G}_\theta(z) \cdot \bar{G}_\theta(z_q), \tag{7}$$

where $\bar{G}_\theta(z)$ is the projected, Hessian-corrected, and unit-normalized gradient for example z, given model parameters θ:

$$\bar{G}_\theta(z) = \frac{G_\theta(z)}{\|G_\theta(z)\|_2} \quad G_\theta(z) = R^{-\frac{1}{2}} P_d \frac{\nabla_\theta \ell(z, \theta)}{\sqrt{V}}.$$

where $V = \mathbb{E}_z[(\nabla_\theta \text{Loss}(z, \theta))^2]$, $R^{-1/2} \in \mathbb{R}^{d \times d}$ is the Hessian matrix using Gaussian-Newton approximation under random projection, and ℓ is defined as negative log likelihood rather than margin function.

Diffusion models. Several works focus on adapting TRAK to diffusion models (Ho et al., 2020a). The most active research direction focuses on the design of the target function $f_\mathcal{A}$ due to the non-trivial adaptation of TRAK designed for classification tasks to diffusion models.

Georgiev et al. (2023) introduces JourneyTrak to attribute the conditional distribution $p_\theta(\mathbf{x} \mid \mathbf{x}_t)$ of image $\mathbf{x}$ generated by diffusion models conditioned on the intermediate latent states at each step t. It takes the training loss $\mathcal{L}_{\text{simple}}(\mathbf{x}, \theta) = \mathbb{E}_{\boldsymbol{\epsilon}, t}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|_2^2\right]$ proposed in DDPM (Ho et al., 2020a) as the target functions. Additionally, Zheng et al. (2023) introduces D-Trak and empirically demonstrates that replacing $f_\mathcal{A}$ with alternative norm-based functions when computing ϕ_m in Equation equation 6, while keeping $\mathcal{L}_{\text{simple}}$ as the loss function $\mathcal{L}$ and using $\mathcal{L}_{\text{simple}}$ for computing Q_k, can significantly improve performance compared to TRAK, in terms of counterfactual evaluation and Linear Datamodeling Scores (LDS). Recently, Lin et al. (2025) points out that the shift of loss function $\mathcal{L}_{\text{simple}}$ fails to capture the distributional changes of generated images, as it conducts an indirect distributional comparison. Therefore, Lin et al. (2025) proposes Diffusion Attribution Score (DAS) by further revising $f_\mathcal{A}$ to be the noise predictor output $\boldsymbol{\epsilon}_\theta(\mathbf{x}, t)$ of the generated sample and directly attributes the KL-divergence between the predicted distributions $D_{\text{KL}}(p_\theta(\mathbf{x}) \| p_{\theta_{-i}}(\mathbf{x}))$, which mitigates the indirect comparison caused by using $\mathcal{L}_{\text{simple}}$ as $f_\mathcal{A}$. Lin et al. (2025) also derives the parameter change for diffusion model trained on $\mathcal{L}_{simple}$, which coincidences with Mlodozeniec et al. (2025) and further improves the LDS. We summarize target functions used in these methods in Table 5.

Multimodal tasks. TRAK can also be applied to attribute multimodal tasks. For example, enabling an understanding of which training data contribute to the image-text pair association. For CLIP models, Park et al. (2023) first propose to reformulate CLIP loss as a classification problem,

$$\ell(x_i, y_i; \theta) = -\log p_1(x_i, y_i; \theta) - \log p_2(x_i, y_i; \theta),$$

where $p_1(x_i, y_i; \theta)$ and $p_2(x_i, y_i; \theta)$ are defined as:

$$p_1(x_i, y_i; \theta) = \frac{\exp(S_{ii})}{\sum_{1 \le j \le n} \exp(S_{ij})}, \quad p_2(x_i, y_i; \theta) = \frac{\exp(S_{ii})}{\sum_{1 \le j \le n} \exp(S_{ji})}.$$

Here, (x_i, y_i) denote an image-text pair and S_{ij} denotes the pairwise similarity score between the i-th image and j-th text embeddings, typically computed using cosine similarity. Then they propose the target function defined as

$$\begin{aligned} f_\mathcal{A}(x_i, y_i; \theta) &= \log \frac{p_1(x_i, y_i; \theta)}{1 - p_1(x_i, y_i; \theta)} + \log \frac{p_2(x_i, y_i; \theta)}{1 - p_2(x_i, y_i; \theta)} \\ &= -\log \sum_{1 \le j \le n} \exp(S_{ij} - S_{ii}) - \log \sum_{1 \le j \le n} \exp(S_{ji} - S_{ii}). \end{aligned}$$

TRAK demonstrates strong empirical performance while significantly reducing the computation and memory cost compared to linear datamodels and original influence functions.

Table 4: Summary of methods, generative models, model behaviors and hyperparameters in the largest experiments. For image generation task, we report n as number of candidate images while for token prediction task, n is taken as the number of tokens in the candidate datasets.

Method	Generative Models	Model Behaviors	p	n	m	k
TRAK	Language Models	Token Prediction	3×10^8	1.5×10^6	10	4000
	CLIP	Multimodal Contrastive Learning	ResNet-50	MS COCO	100	20000
JourneyTrak	Diffusion Models	Image Generation	3.57×10^7	CIFAR-10	100	16384
D-Trak	Diffusion Models	Image Generation	3.57×10^7	5×10^3	1	32768
			2.55×10^6	1.25×10^4	1	32768
DAS	Diffusion Models	Image Generation	11.9×10^7	5×10^3	1	32768
DsDM	Language Models	Token Prediction	1.25×10^8	6×10^9	4	16384

Table 5: Summary of methods and their target functions. For language model tasks, T refers to the context length. For D-Trak method, we specify the target function used to compute the gradient term in TRAK. $\bar{p}$ defines the mean probability that the model correctly predicts the masked token or the next token in example x. We also list the Q matrix in the implementation for TRAK-based methods. † marks terms using negative output-to-loss gradient. We summarize the implementation of Q matrix in Table.

Method	Target Function	Formulation	Q Matrix
TRAK	Multi-class margin	$\log \frac{p(x)}{1-p(x)}$	$\mathrm{diag}(\{1-p(x)\})^\dagger$
TRAK - Masked LM	Multi-class margin	$\sum_{j \in \text{masked tokens}} \log\left(\frac{p(x_j \mid x_{-j};\theta)}{1-p(x_j \mid x_{-j};\theta)}\right)$	$\mathrm{diag}(\{1-\bar{p}(x)\})^\dagger$
TRAK - CLIP	Multi-class margin	$-\log \sum_{1\le j\le n} \exp(S_{ij}-S_{ii}) - \log \sum_{1\le j\le n} \exp(S_{ji}-S_{ii})$	$\mathrm{diag}(\{1-\frac{p_1(x,y)+p_2(x,y)}{2}\})^\dagger$
JourneyTrak	DDPM loss function	$\mathcal{L}_{\text{simple}} = \frac{1}{k}\sum_{i=1}^{k} \left\|\boldsymbol{\epsilon}_i - \boldsymbol{\epsilon}_\theta\left(\sqrt{\bar{\alpha}_t}\hat{\mathbf{x}}_0^{(t)} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_i, t\right)\right\|_2^2$	$\mathbf{I}$
D-Trak	Norm-based loss function	$\mathcal{L}_{\text{square}}(\mathbf{x},\theta) = \mathbb{E}_{t,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon}_\theta(\mathbf{x}_t,t)\right\|_2^2\right]$	$\mathbf{I}$
		$\mathcal{L}_{\text{Avg}}(\mathbf{x},\theta) = \mathbb{E}_{t,\boldsymbol{\epsilon}}\left[\text{Avg}(\boldsymbol{\epsilon}_\theta(\mathbf{x}_t,t))\right]$	$\mathbf{I}$
		$\mathcal{L}_{p\text{-norm}}(\mathbf{x},\theta) = \mathbb{E}_{t,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon}_\theta(\mathbf{x}_t,t)\right\|_p\right]$	$\mathbf{I}$
DAS	Noise prediction	$\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$	-
DsDm	Multi-class margin	$\sum_{j=2}^{T} \log\left(\frac{p(x_j \mid x_{<j};\theta)}{1-p(x_j \mid x_{<j};\theta)}\right)$	$\mathrm{diag}(\{1-\bar{p}(x)\})^\dagger$
TrackStar	Negative log-likelihood	$-\sum_{j=2}^{T} \log p(x_j \mid x_{<j};\theta)$	$\mathbf{I}$

2.2.5 Key Takeaways and Future Directions

Recent advances in scaling influence functions for large models have converged on a common recipe: applying gradient projection techniques (Park et al., 2023; Chang et al., 2025) to reduce dimensionality, combined with matrix factorization (Grosse et al., 2023) or other Hessian approximation techniques (Kwon et al., 2024; Mlodozeniec et al., 2025) to make computations tractable. While these modifications have enabled influence estimation in settings that were previously infeasible, rigorous ablations are lacking; for instance, it remains unclear whether FIM or GGN yields better empirical performance. More fundamentally, after these efficiency-oriented modifications, it is unclear whether the resulting metric still approximates LOO or instead tracks a different quantity. Addressing these questions represents an important direction for future work.

2.3 Category: Weighted Marginal Contribution Methods

Another line of research quantifies influence through the *weighted marginal contribution* of training samples. In this framework, a sample's marginal impact is evaluated over subsets of the training dataset, with each subset assigned a specific weight. Unlike simple LOO which assesses a sample's importance only in relation to the full training dataset, this weighted aggregation captures more nuanced interactions between data points, leading to a more comprehensive and robust estimation of sample influences. Notable methods in this category include Data Shapley (Ghorbani and Zou, 2019; Jia et al., 2019b), Beta Shapley (Kwon and Zou, 2022), and Data Banzhaf (Wang and Jia, 2023a), which we collectively refer to as *probabilistic values* (Dubey and Weber, 1977) (the formal probabilistic interpretation will be discussed in Equation (8)). In Section 2.3.1, we first review these key concepts and discuss their respective strengths and limitations. Next, we explore efficient methods for computing probabilistic values in Section 2.3.2. Finally, we discuss the limitations of this research direction, particularly in the context of generative AI, and outline potential future work in Section 2.3.3.

We note that methods in this category are closely related to game theory, and many works are directly motivated by the desire to establish a principled framework for compensating data holders (Ghorbani and Zou, 2019). As a result, the term *data valuation* appears more frequently than data attribution, although they share the same technical meaning. Throughout this section, we use data valuation and data attribution interchangeably.

Notation. Given a training set $D = \{z_i\}_{i=1}^n$, let $S \subset D$ be a subset of the training data, and let θ_S denote the model parameters obtained by training on S. A utility function $U : 2^D \to \mathbb{R}$ quantifies the performance of the model trained on S, typically via an external validation set[10] $V = \{v_i\}_{i=1}^m$. When the context is clear, we write $U(S)$ as the evaluation of S and ignore V. We denote the marginal contribution of a data point z_i to a subset S as $\Delta(z_i, U, S) = U(S \cup \{z_i\}) - U(S)$. For instance, the LOO error of z_i can be expressed as $\Delta(z_i, U, D \setminus \{z_i\})$. Finally, we denote a *value function* (Jia et al., 2019b) by ϕ^U, which assigns a value ϕ_i^U to each z_i, and omit the superscript when the context is clear.

2.3.1 Notions of Probabilistic Values

Data Shapley. (Ghorbani and Zou, 2019; Jia et al., 2019b) is one of the earliest principled frameworks for data valuation (Sim et al., 2022) in machine learning. It is based on the Shapley value (Shapley, 1953), a concept originating from cooperative game theory. Formally, the Shapley value for z_i is computed as the weighted average of its marginal contributions over all possible subsets $S \subseteq D \setminus \{z_i\}$

$$\phi_i = \frac{1}{n} \sum_{k=1}^{n} \binom{n-1}{k-1}^{-1} \sum_{S \subseteq D \setminus \{z_i\}, |S|=k-1} \Delta(z_i, U, S).$$

Ghorbani and Zou (2019) argue that Data Shapley offers an *equitable* division of value, whereas LOO does not. More specifically, Data Shapley is recognized as the *unique* value function that simultaneously satisfies the following four axioms, which are typically desirable for data valuation:

1. **Null Player**: If adding z_i has no effect on any subset, it should receive a value of zero. Formally, if $U(S \cup \{z_i\}) = U(S), \forall S \subseteq D \setminus \{z_i\}$, then $\phi_i = 0$.
2. **Symmetry**: If z_i and z_j contribute equally to all subsets, they should receive the same value. Formally, if $U(S \cup \{z_i\}) = U(S \cup \{z_j\}), \forall S \subseteq D \setminus \{z_i, z_j\}$, then $\phi_i = \phi_j$.
3. **Additivity**: The values are additive under the utility functions. Formally, for two utility functions U_1, U_2 and $\alpha_1, \alpha_2 \in \mathbb{R}$, $\phi_i^{\alpha_1 U_1 + \alpha_2 U_2} = \phi_i^{\alpha_1 U_1} + \phi_i^{\alpha_2 U_2}$.
4. **Efficiency**: The value of the entire dataset is completely distributed among all samples. Formally, $\sum_{i=1}^n \phi_i = U(D)$.

Since its inception, Data Shapley has been applied to a variety of machine learning tasks (Shim et al., 2021; Liu et al., 2022; Rozemberczki et al., 2022) and has motivated several generalizations, including distributional Shapley (Ghorbani et al., 2020; Kwon et al., 2021; Li and Yu, 2024a) and class-wise Shapley (Schoch et al., 2022). However, despite these promising advances, Data Shapley faces several issues in both its theoretical formulation and practical implementation. These limitations have motivated the development of alternative data valuation schemes, which we discuss below.

Beta Shapley. Kwon and Zou (2022) argue that the efficiency axiom is not necessary for machine learning for two main reasons. First, data valuation based on utility functions defined on some test set is typically not directly tied to monetization. Second, the ordering of samples is often sufficiently informative for many applications (e.g., data selection or mislabeled data detection), so that knowing the exact absolute values adds little extra value. Based on these arguments, they propose removing the efficiency axiom, leading to a broader class of value functions known as *semivalues* (Dubey and Weber, 1977; Dubey et al., 1981). Semivalues satisfy the other three axioms, even though they may not always adhere to the efficiency axiom. Formally,

Definition 2.1 (Semivalue)**.** A value function ϕ is considered a semivalue if and only if there exists a set of weights $\{\alpha_k^{(n)}, k = 1, \ldots, n\}$ satisfying $\sum_{k=1}^n \binom{n-1}{k-1} \alpha_k^{(n)} = 1$, such that the value function ϕ can be expressed as follows:

$$\phi_i =: \sum_{k=1}^{n} \alpha_k^{(n)} \sum_{S \subseteq N \setminus \{z_i\}, |S|=k-1} \Delta(z_i, U, S). \tag{8}$$

[10]The choice (e.g., quality or representativeness) of the validation set can significantly impact the resulting data values. Motivated by this, several works have addressed the challenge of data valuation in the absence of reliable validation datasets (Kwon and Zou, 2023; Lin et al., 2024c; Jahagirdar et al., 2024).

The condition $\sum_{k=1}^{n}\binom{n-1}{k-1}\alpha_k^{(n)}=1$ naturally induces a probability distribution over all subsets $S\subseteq N\setminus\{z_i\}$, where subsets of the same size receive equal weight. Under this distribution, ϕ_i represents the *expected* contribution of z_i when joining a randomly sampled subset S. This probabilistic perspective also motivates the term *probabilistic value*. Kwon and Zou (2022) further propose a simple characterization of the coefficients: for any probability measure ξ on $[0,1]$, a set of weights $\{\alpha_k^{(n)}\}_{k\in[n]}$ defined by

$$\alpha_k^{(n)}=\int_0^1 t^{k-1}(1-t)^{n-k}\mathrm{d}\xi(t) \tag{9}$$

satisfies the condition $\sum_{k=1}^{n}\binom{n-1}{k-1}\alpha_k^{(n)}=1$, making the corresponding ϕ a semivalue. Mathematically, Equation (9) is equivalent to sampling each data point independently with a probability t, where t is drawn from the probability distribution ξ. This formulation also arises in the Average Marginal Effect (AME) (Lin et al., 2022), which is motivated by measuring multiple treatment effects using randomized experiments in the literature of causal inference (Imbens and Rubin, 2015; Egami and Imai, 2019).

Both Data Shapley and LOO can be viewed as specific instances of a semivalue, in particular as instantiations of Equation (9). Concretely, Data Shapley is obtained by setting $\xi=\mathrm{Unif}([0,1])$, which leads to

$$\alpha_k^{(n)}=\frac{1}{n}\binom{n-1}{k-1}^{-1}.$$

In contrast, LOO is obtained by setting $\xi=\delta_1$ (the Dirac measure concentrated at $t=1$, whose cumulative distribution function is $\mathbb{1}(t\geq 1)$), yielding

$$\alpha_k^{(n)}=\mathbb{1}[k=n].$$

Beta Shapley, on the other hand, employs the beta distribution $\xi=\mathrm{Beta}(\alpha,\beta)$, which results in

$$\alpha_k^{(n)}=\frac{\mathrm{Beta}(\alpha+n-k,\beta+k-1)}{\mathrm{Beta}(\alpha,\beta)}.$$

We note that Beta Shapley is a more general and flexible notion that encapsulates both Data Shapley and LOO as special cases: Data Shapley is recovered by setting $(\alpha,\beta)=(1,1)$, while LOO is obtained by setting $\alpha=1$ and taking $\beta\to\infty$. More generally, α and β can be interpreted as the weights placed on subsets of small and large cardinality, respectively. The main thesis of Kwon and Zou (2022) is that the marginal contributions calculated at sufficiently large subsets is often negligible, making it difficult to distinguish samples that significantly enhance model performance. Therefore, they recommend choosing $\alpha>\beta$, which effectively downweights noisy contributions from larger subsets and provides a more robust estimation.

Data Banzhaf. Wang and Jia (2023a) observe that in machine learning, the computation of utility scores $U(S)$ can be noisy due to the inherent randomness in weight initialization and minibatch sampling. This further makes the calculated values ϕ_i's unreliable; as a matter of fact, different runs of the same data valuation algorithm may produce inconsistent value rankings. To quantify the brittleness, they introduce the concept of *safety margin*, defined as the maximum perturbation in utility scores that can be tolerated while preserving the relative value ordering of every pair of data points. They then propose the Data Banzhaf framework based on the Banzhaf value (Banzhaf III, 1964), a classic notion from cooperative game theory and an instantiation of a semivalue. Formally, it corresponds to setting $\xi=\delta_{0.5}$ in Equation (9), resulting in

$$\alpha_k^{(n)}=\frac{1}{2^n}$$

for all k. In other words, Data Banzhaf assigns *uniform* weight to all subsets, regardless of their sizes. Wang and Jia (2023a) further demonstrate that Data Banzhaf achieves the largest safety margin among all semivalues, enabling it to *robustly* distinguish data quality in the presence of noise.

Weighted Banzhaf values. Li and Yu (2023) demonstrate that if one assumes a specific *structure* for the stochasticity (rather than relying on the safety margin, which is a *worst-case* robustness notion), Data Banzhaf may not be the most robust semivalue. Concretely, they show that when the stochasticity is parameterized by Kronecker noise, the uniquely most robust semivalue which minimizes the worst-case entropy belongs to the family of weighted Banzhaf values (w-weighted Banzhaf value), a generalized form of the Banzhaf value. Formally, this corresponds to setting $\xi=\delta_w$ in Equation (9), yielding

$$\alpha_k^{(n)}=w^{k-1}(1-w)^{n-k}.$$

Notably, 0.5-weighted Banzhaf value exactly recovers the standard Banzhaf value used in Data Banzhaf.

2.3.2 Efficient Computations of Probabilistic Values

Having reviewed the definitions of various probabilistic values in Section 2.3.1, we now turn our attention to their computation. Taking Shapley values as an example, computing them *exactly* requires evaluating U on 2^n different subsets of the training data, each corresponding to training a different model. This exponential complexity renders an exact calculation infeasible even for moderately sized datasets. In fact, Deng and Papadimitriou (1994) show that computing these values is NP-hard. In what follows, we discuss two main directions to efficiently compute probabilistic values: reducing the number of evaluations required and lowering the computational cost per evaluation.[11]

Table 6: Summary of applicability and convergence rates for various methods of estimating probabilistic values. Theoretically, OFA (Li and Yu, 2024b) achieves the best-known convergence rate for all probabilistic values on average. Empirically, Li and Yu (2024b) report that OFA generally outperforms other methods, although the complement estimator and MSR estimator perform comparably to OFA for the Shapley and Banzhaf values, respectively. For practitioners, since the target probabilistic value is typically known in advance, we recommend using the complement and MSR estimators for their simpler implementation.

Method	Applicability	Convergence rate	Remark
Permutation Sampling (Castro et al., 2009)	Shapley	$O(n^2 \log n)$	—
Group Testing (Jia et al., 2019b)	Shapley	$O(n(\log n)^2)$	Empirically underperforms
Weighted Sampling Lift (Kwon and Zou, 2022)	All	$O(n^2 \log n)$	—
Sparse Regression (Lin et al., 2022)	Partial	$O(k \log n)$	Does not apply to Shapley value
Complement Estimator (Zhang et al., 2023a)	Shapley	Unknown	—
MSR (Wang and Jia, 2023a)	Banzhaf	$O(n \log n)$	—
OFA (Li and Yu, 2024b)	All	$O(n \log n)$	—

Reducing the number of evaluations. Since the definitions of probabilistic values take the form of a weighted sum, a natural strategy is to employ Monte Carlo sampling to approximate them, which inherently ensures *unbiasedness*. The central question for these approximation methods is: *how many evaluations are required to ensure that the estimated probabilistic values are close to the ground truth?* Here, closeness is typically characterized via the (ε, δ)-approximation:

$$\Pr(\|\hat{\phi} - \phi\|_2 \leq \varepsilon) \geq 1 - \delta,$$

where $\hat{\phi}$ denotes the estimated score vector (*estimator* in short). We refer to the number of evaluations required (denoted by T) as a function of n as the *convergence rate* for each method, which will be the main quantity of interest. In what follows, we review several representative approximation methods, discuss their applicability and convergence rates, and summarize them in Table 6. Before that, we make an additional comment.

Remark 2.2. Approximating probabilistic values, particularly the Shapley value, is a research problem that extends beyond data attribution. For instance, it has been extensively studied in the context of *feature attribution* (Lundberg and Lee, 2017), and many of those methods directly transfer to data attribution. Nevertheless, to avoid confusion, we focus our discussion on works that explicitly address data attribution, and refer readers to Li and Yu (2024a,b) for a more comprehensive discussion, which includes methods such as KernelSHAP (Lundberg and Lee, 2017; Covert and Lee, 2021), FastSHAP (Jethani et al., 2022), SHAP-IQ (Fumagalli et al., 2024), ARM (Kolpaczki et al., 2024), and others.

Permutation sampling. One of the most widely used estimator for Shapley value is based on permutation sampling, proposed by Castro et al. (2009). The key observation is that the Shapley value can be equivalently formulated as

$$\phi_i = \frac{1}{n!} \sum_{\pi \in \Pi} \Delta(z_i, U, \mathcal{P}^i(\pi)),$$

where Π contains all permutations of $[n]$ and $\mathcal{P}^i(\pi)$ is the subset that contains all players preceding i in π. Based on this equation, the permutation estimator is naturally defined as

$$\hat{\phi}_i = \frac{1}{T} \sum_{j=1}^{T} \Delta(z_i, U, \mathcal{P}^i(\pi_j)),$$

with $\{\pi_j\}_{j=1}^T$ being uniformly sampled (with replacement) from Π. Assume that U is bounded, an application of Hoeffding's bound suggests that $O(n \log n)$ samples are required to ensure that $\hat{\phi}_i$ is close to ϕ_i, implying a total requirement of $T = O(n^2 \log n)$ evaluations.

[11] We include the cost of model training in evaluation.

Permutation sampling is less efficient because the marginal contribution $\Delta(z_i, U, S_j)$ (which requires two evaluations) is only used to update $\hat{\phi}_i$, thereby introducing an extra factor of n in the convergence rate. Nevertheless, due to its simplicity, permutation sampling has served as a cornerstone for many subsequent works. For example, Ghorbani and Zou (2019) propose Truncated Monte Carlo Shapley (TMC-Shapley), which truncates the calculation of marginal contributions in a sampled permutation once the utility evaluation is close to that of the full training set. While TMC-Shapley is more efficient than the standard permutation sampling method, it sacrifices unbiasedness. Jia et al. (2019b) leverage permutation sampling to construct the label vector within a compressed sensing framework, achieving a convergence rate of $O(n \log \log n)$. It is important to note that this method relies on the assumption that the ground truth Shapley values are "approximately sparse" (i.e., most ϕ_i's are close to their mean).

Group testing. Jia et al. (2019b) propose an estimator based on group testing (Du and Hwang, 1999). The key idea is to estimate the Shapley differences between all data pairs instead of the Shapley values directly

$$\phi_i - \phi_j = \frac{1}{n-1} \sum_{S \subseteq D \setminus \{z_i, z_j\}} \frac{1}{\binom{n-2}{|S|}} [U(S \cup \{z_i\}) - U(S \cup \{z_j\})],$$

followed by solving a feasibility problem with an additional constraint imposed by the efficiency axiom to obtain the $\hat{\phi}_i$'s. This approach achieves a convergence rate of $O(n(\log n)^2)$, saving a factor of n compared to the permutation sampling approach, thanks to increased *sample reuse*: each utility evaluation contributes to the estimation of the Shapley values for all samples. Wang and Jia (2023b) provide a refined analysis of the group testing method and also propose an improved estimator using the dummy player technique (we refer interested readers to their work for details). On the other hand, they make two key observations: 1) group testing does not achieve *maximum sample reuse* (MSR), a concept that will be discussed in detail shortly; 2) empirically, group testing does not significantly outperform permutation sampling, despite having a faster convergence rate.

(Weighted) sampling lift. Equation (8) provides a natural probabilistic interpretation. First, the outer summation corresponds to a discrete probability distribution over the *cardinality* of the subset, with the probability of a subset having cardinality k given by $\binom{n-1}{k-1}\alpha_k^{(n)}$. Second, the inner summation represents a uniform distribution over all subsets of cardinality k. This idea underlies sampling lift (Moehlea et al., 2022), where the estimator is defined as

$$\hat{\phi}_i = \frac{1}{T} \sum_{j=1}^{T} \Delta(z_i, U, S_j),$$

and the $\{S_j\}_{j=1}^T$ are sampled independently from the two-step process above: first, draw the subset size from the discrete probability distribution, then uniformly sample a subset of that size.

Kwon and Zou (2022) extend this idea with a weighted version of sampling lift. Specifically, they reformulate Equation (8) as

$$\phi_i = \frac{1}{n} \sum_{k=1}^{n} n\alpha_k^{(n)} \binom{n-1}{k-1} \left[\frac{1}{\binom{n-1}{k-1}} \sum_{\substack{S \subseteq D \setminus \{z_i\} \\ |S|=k-1}} \Delta(z_i, U, S) \right],$$

and propose the estimator

$$\hat{\phi}_i = \frac{1}{T} \sum_{j=1}^{T} n\alpha_{|S_j|}^{(n)} \binom{n-1}{|S_j|-1} \Delta(z_i, U, S_j),$$

where the sampling process for $\{S_j\}_{j=1}^T$ is modified so that, in the first step, the cardinality of S_j is sampled uniformly from $[n]$. For the Shapley value, the coefficients $n\alpha_{|S_j|}^{(n)}\binom{n-1}{|S_j|-1}$ simplify to 1. However, for general probabilistic values, computing these coefficients is numerically unstable for large n due to the combinatorial factors.

(Weighted) sampling lift achieves a convergence rate of $T = O(n^2 \log n)$, with the proof strategy mirroring that of the permutation sampling approach (e.g., see Proposition 5 in Li and Yu (2024a)). Importantly, since the estimator is directly derived from Equation (8), it can be applied to any probabilistic values.

Sparse regression. Lin et al. (2022) observe that a subfamily of probabilistic values can be formulated as the optimal solution to a least squares problem. This methodology also underlines earlier works in feature attribution such as KernelSHAP (Lundberg and Lee, 2017) and more recent works such as GELS (Li and Yu, 2024a). Specifically, when ξ (as defined in Equation (9)) satisfies $C_\xi =: \int_0^1 \frac{1}{w(1-w)} \mathrm{d}\xi(w) < \infty$, we have

$$\phi = \underset{v \in \mathbb{R}^N}{\arg\min}\, \mathbb{E}_S \left[(Y(S) - X(S)^\top v)^2 \right].$$

Here, $Y(S) = U(S)$ and $X(S)_i$ (the i-th coordinate of $X(S)$) equals $\frac{1}{C_\xi w}$ if $i \in S$ and otherwise $-\frac{1}{C_\xi(1-w)}$. The random subset S is generated by first sampling $w \in [0,1]$ from ξ, and then including each sample independently with probability w. The final estimator is given by

$$\hat{\phi}_i = \arg\min_{v \in \mathbb{R}^N} \frac{1}{T} \sum_{j=1}^{T} \left(Y(S_j) - X(S_j)^\top v\right)^2 + \lambda \|v\|_1,$$

where an L_1-regularization term is added to perform sparse regression (i.e., LASSO (Tibshirani, 1996)). Assuming $\|\phi\|_0 = k$, Lin et al. (2022) show that the convergence rate of this estimator is $O(k \log n)$, a significant improvement over previous methods. However, the requirement that C_ξ be finite precludes the application of this method to the Shapley value: in particular, when $\xi = \text{Unif}([0,1])$, we have $\int_0^1 \frac{1}{w(1-w)} \mathrm{d}w = \int_0^1 \left(\frac{1}{w} + \frac{1}{1-w}\right) \mathrm{d}w = \infty$.

Complementary contribution. Another equivalent formulation of the Shapley value is based on the complementary formula (Harsanyi, 1982). Instead of expressing ϕ_i as a weighted summation of marginal contributions, it is written as a weighted summation of complementary contributions (CC), which are formally defined as $CC(S) = U(S) - U(D \backslash S)$ for a subset S. The complementary formula states that

$$\phi_i = \frac{1}{n} \sum_{S \subseteq D \backslash \{z_i\}} \binom{n-1}{|S|}^{-1} CC(S \cup \{z_i\}).$$

Based on this, Zhang et al. (2023a) propose the following complement estimator:

$$\hat{\phi}_i = \frac{1}{n} \sum_{k=1}^{n} \frac{1}{T_{i,k}} \sum_{j=1}^{T} CC(S_j) \left(\mathbb{1}(i \in S_j,\, |S_j| = k) - \mathbb{1}(i \notin S_j,\, n - |S_j| = k)\right),$$

where $T_{i,k} = \sum_{j=1}^{T} \left(\mathbb{1}(i \in S_j,\, |S_j| = k) + \mathbb{1}(i \notin S_j,\, n - |S_j| = k)\right)$, and $\{S_j\}_{j=1}^T$ are sampled independently by first sampling the subset size uniformly from $[n]$, and then uniformly drawing a subset of that size. The main advantage of this estimator is that each sample appears either in a subset S or in its complement, so a single utility evaluation can be leveraged to update the estimates for all samples. In fact, as suggested by Li and Yu (2024b), the complement estimator adheres to the principle of MSR, which will be discussed in the next method. Consequently, although there is no formal convergence guarantee, it is reported to have the best empirical performance for estimating the Shapley value (Li and Yu, 2024b).

Maximum sample reuse. The principle of MSR has appeared in prior works, such as group testing and the complement estimator, and was formally introduced in Wang and Jia (2023a) to efficiently approximate the Banzhaf value. Formally, MSR requires that each evaluation is used to update the estimates for all ϕ_i's. To achieve this, Wang and Jia (2023a) rewrite the Banzhaf value as

$$\phi_i = \mathbb{E}_{S|z_i \in S}[U(S)] - \mathbb{E}_{S|z_i \notin S}[U(S)], \tag{10}$$

where S is uniformly distributed over all subsets of D. They then propose the following MSR estimator:

$$\hat{\phi}_i = \frac{1}{T_i^+} \sum_{j=1}^{T} U(S_j) \mathbb{1}(i \in S_j) - \frac{1}{T_i^-} \sum_{j=1}^{T} U(S_j) \mathbb{1}(i \notin S_j),$$

where $T_i^+ = \sum_{j=1}^{T} \mathbb{1}(i \in S_j)$ and $T_i^- = \sum_{j=1}^{T} \mathbb{1}(i \notin S_j)$, and $\{S_j\}_{j=1}^T$ are sampled independently from the uniform distribution over all subsets of D. Thanks to MSR, this estimator achieves a convergence rate of $O(n \log n)$.

While this estimator can naturally be adapted to the weighted Banzhaf value, it cannot be applied to the Shapley value because it is impossible to construct a suitable distribution over D to express Shapley values in the form of Equation (10) (Wang and Jia, 2023a; Li and Yu, 2024a). We note that this is a limitation of marginal contributions: as demonstrated in the previous method, the complement estimator achieves MSR by using complementary contributions instead.

One-sample-fits-all. Most of the estimators discussed thus far, with the exception of (weighted) sampling lift, cannot be applied to all probabilistic values. However, weighted sampling lift not only has a suboptimal convergence rate $O(n^2 \log n)$, but its convergence can be further degraded by the coefficients $n\alpha_{|S_j|}^{(n)} \binom{n-1}{|S_j|-1}$ (referred to as *amplifying scalars*). To address this challenge, Li and Yu (2024b) introduce the One-sample-Fits-All (OFA) framework, which simultaneously satisfies the following desirable properties: 1) it requires sampling subsets only once to approximate

all probabilistic values; 2) it adheres to the principle of MSR; and 3) it avoids the amplifying scalars inherent in the weighted sampling lift estimator. The framework is based on the following formulation of probabilistic values:

$$\phi_i = \sum_{k=1}^{n} \binom{n-1}{k-1} \alpha_k^{(n)} \cdot \left(\mathbb{E}_{z_i \in S, |S|=k}[U(S)] - \mathbb{E}_{z_i \notin S, |S|=k-1}[U(S)] \right),$$

where each expectation is taken with respect to the corresponding uniform distribution. The two expectations, denoted as $\hat{\phi}_{i,s}^{+} = \mathbb{E}_{z_i \in S, |S|=k}[U(S)]$ and $\hat{\phi}_{i,s}^{-} = \mathbb{E}_{z_i \notin S, |S|=k-1}[U(S)]$, are estimated using the two-step process of the sampling lift estimator. Furthermore, by optimizing the discrete probability distribution over the cardinality of subsets, OFA achieves a convergence rate of $O(n \log n)$ for all probabilistic values on average.

Lowering the cost per evaluation. Despite significant progress along the previous direction, the minimum requirement of utility evaluation remains at least linear in the number of training samples, which is generally impractical for training large-scale models *from scratch*. In the following, we discuss a complementary approach that aims to address this challenge by lowering the computational cost per evaluation. This reduction can be achieved either by employing techniques to decrease the training cost directly, or by substituting the entire training process with a lighter, more computationally efficient *proxy* model[12].

Reducing training cost. Building upon the framework of permutation sampling, Ghorbani and Zou (2019) introduce Gradient Shapley (G-Shapley), which approximates a fully trained model by training for a single epoch with a batch size of 1 (this is referred to as an *incrementally trainable* model in Jia et al. (2019b)) and then computing the marginal contributions as data is sequentially added. Lu et al. (2025) propose Sparsified Fine-Tuning Shapley (Sparsified-FT Shapley) for diffusion models. This method involves pruning a pre-trained diffusion model and fine-tuning it on the full training set to produce a sparsified yet performant model, which is then used to fine-tune on subsets of the training data for utility evaluation. Both approaches significantly reduce the computational cost compared to full retraining.

KNN as a proxy. Jia et al. (2019a) derive a closed-form expression for the Shapley value in the context of the *K-Nearest Neighbors* (KNN) classifier. Specifically, they introduce a KNN utility function that quantifies the impact of a subset S on KNN classification accuracy:

$$U(S) = \frac{1}{K} \sum_{k=1}^{\min\{K,|S|\}} \mathbb{1}[y_{\alpha_k(S)} = y_{\text{val}}],$$

where $(x_{\text{val}}, y_{\text{val}})$ represents a validation sample and $\alpha_k(S)$ denotes the indices of the training data in S sorted by increasing distance to x_{val}. Building on this utility function, they derive a recursive algorithm that computes the exact Shapley values in $O(n \log n)$ time:

$$\phi_{\alpha_n} = \frac{1}{n} \mathbb{1}[y_{\alpha_n} = y_{\text{val}}],$$
$$\phi_{\alpha_i} = \phi_{\alpha_{i+1}} + \frac{\mathbb{1}[y_{\alpha_i} = y_{\text{val}}] - \mathbb{1}[y_{\alpha_{i+1}} = y_{\text{val}}]}{K} \frac{\min\{K, i\}}{i}, i = n-1, \cdots, 1.$$

More importantly, Jia et al. (2019a) observe that this framework for calculating Shapley values for KNN classifiers can be extended to deep neural networks. Specifically, they propose to train a KNN classifier based on the embeddings (e.g., extracted from deeper layers of the network or pre-trained feature extractors) of each x_i along with their corresponding labels y_i, and then perform data attribution in the latent space. This effectively reduces the problem of data attribution for deep neural networks to that of KNN classification. This approach, which avoids the need to repeatedly train complex models, has been recognized in follow-up works as one of the most practical and scalable techniques for large-scale data valuation (Jia et al., 2021; Pandl et al., 2021; Karlaš et al., 2024).

NTK as a proxy. The neural tangent kernel (NTK) (Jacot et al., 2018) is a powerful analytical tool in deep learning theory, used to investigate the optimization and generalization of (deep) neural networks. It posits that with a suitable scale of random initialization, the training of a sufficiently wide neural network remains close to its initialization, effectively reducing the learning dynamics to a kernel regression characterized by NTK. In the context of probabilistic values, NTK enables an approximate evaluation of the utility function $U(S)$ without the need for full training, thereby significantly lowering computational costs. For instance, Wu et al. (2022b) introduce *data valuation at initialization* (DaVinz), leveraging an off-the-shelf generalization bound from Arora et al. (2019) to compute $U(S)$. Similarly, Wang et al. (2024g) propose *fine-tuning-free Shapley value* (FreeShap), which employs the empirical NTK (Malladi et al., 2023) to characterize $U(S)$. We comment that NTK is particular useful in the fine-tuning regime where the model is expected to remain close to the pretrained checkpoint.

[12]We focus on general-purpose proxy models and omit those with a relatively narrow scope, such as Data-OOB (Kwon and Zou, 2023), which is only applicable to bagging models.

2.3.3 Key Takeaways and Future Directions

The weighted marginal contribution method builds upon the widely recognized Shapley value framework, with Data Shapley remaining the most popular notion of probabilistic value for training data attribution. Despite significant progress in this area, this approach is generally less scalable than influence functions (as discussed in Section 2.2) because it requires at least a linear number of model trainings relative to the dataset size. In fact, most algorithms discussed in this section, except those based on training-free proxy models, are not directly applicable in the context of generative AI. Nonetheless, its strong theoretical foundations, especially its satisfaction of the four desirable axioms, make it a compelling alternative in scenarios where attribution is needed at a higher level, such as domains (Kang et al., 2024; Liu et al., 2025) or copyright owners (Wang et al., 2024c). This is particularly relevant in emerging applications within the data economy (Sestino et al., 2025) and data markets (Zhang et al., 2024). We envision future work focusing on more efficient approximations of Data Shapley and related concepts tailored to these applications.

Remark 2.3 (**Preservation of axioms under Monte-Carlo approximation**). Among the four classical axioms that characterize the Shapley value, the symmetry axiom is often viewed as the key *fairness* requirement: equally useful points should receive equal rewards. When the exact Shapley value is approximated with Monte-Carlo (MC) sampling, however, MC estimates satisfy Symmetry only *in expectation*. With a finite sample budget, two exchangeable points typically obtain slightly different estimates because the Monte-Carlo noise that perturbs their marginal contributions is independent. There is an emerging line of work developing Shapley value approximation methods that provably preserve most of the Shapley axioms. Wang et al. (2024d) developed a deterministic approximation algorithm for weighted KNN-Shapley that preserves the null player and symmetry axioms exactly.

2.4 Category: Training Dynamics Methods

This category of methods leverages information collected throughout the training process to assess data influence, an approach we refer to as *training dynamics* data attribution. The literature in this area generally divides into two groups: *tracing gradient descent* (`TracIn`) and *trajectory-specific Leave-One-Out* (`TSLOO`).

1. *Tracing gradient descent*. Initiated by Pruthi et al. (2020), this approach, referred to as `TracIn`, estimates the influence of training data by approximating the cumulative change in model outputs during training.
2. *Trajectory-specific LOO*. Originally introduced by Hara et al. (2019) under the name `SGD-influence` (or counterfactual SGD), this method was later reframed as `TSLOO` in a follow-up study (Wang et al., 2025b) to more accurately reflect its goal of dynamically approximating leave-one-out. Specifically, it quantifies the influence of removing a single training sample on both the model parameters and outputs over the course of training by building a recursive relationship between consecutive iterations.

Notation. The loss incurred on a single data point is denoted by $\ell(z;\theta)$, where $\theta \in \mathbb{R}^p$ represents the model parameters and $z \in D$, given a training set $D = \{z_i\}_{i=1}^n$. We also define $f(z_{\text{test}};\theta)$ as the target function and z_{test} as a test sample.

2.4.1 Tracing Gradient Descent

Derivation of tracing gradient descent. We review the basic derivation of tracing gradient descent method family, particularly highlight the connection between `TracIn-Ideal` and its first-order approximation `TracIn`.

`TracIn-Ideal`. The high-level idea of `TracIn-Ideal` is accumulating the change in model outputs caused by optimizing on a training sample throughout the training process. The idealized influence of a particular training sample z on a given test sample z_{test} is defined as the total model output $f(z_{\text{test}};\theta)$ change induced by the whole training process whenever z is trained on. The definition can be formalized as:

$$\texttt{TracIn-Ideal}(z, z_{\text{test}}) \triangleq \sum_{t:z_t=z} (f(z_{\text{test}};\theta_t) - f(z_{\text{test}};\theta_{t+1})), \tag{11}$$

where $t : z_t = z$ indicates the iteration where z is taken as the training sample. Equation 11 is for vanilla SGD where parameters optimization is done on one training sample at a time. This ideal definition (Pruthi et al., 2020) is the goal to be approximated by tracing gradient descent method family.

`TracIn`. To make the tracing computationally tractable, `TracIn` uses first-order Taylor expansion to approximate the idealized influence, and converts the Equation 11 into the following inner-product formulation. The formula is also adapted to mini-batch training.

$$\texttt{TracIn}(z, z_{\text{test}}) \triangleq \frac{1}{|B_t|} \sum_{t:z\in B_t} \eta_t \langle \nabla_\theta f(z_{\text{test}}; \theta_t), \nabla_\theta \ell(z; \theta_t) \rangle, \tag{12}$$

It assumes that parameters are updated by an SGD optimization process with training batch B_t and learning rate η_t at iteration t. Most of the developments on tracing gradient decent family are based on this first-order approximation.

Challenges. Among recent studies, tracing gradient decent family can be primarily categorized into two groups: *offline tracing algorithms* and *online tracing algorithms*. *Offline tracing algorithms* (Pruthi et al., 2020; Lin et al., 2024b; Xie et al., 2024a; Yeh et al., 2022) present the traditional implementations to `TracIn`, calculating attribution results **after** model training, typically leveraging some cached training checkpoints. *Online tracing algorithms* (Wang et al., 2025a, 2024e) compute the attribution results of each training iteration **during** each training iteration alongside model updates, utilizing access to richer training dynamics such as backpropagation information.

The scalability of Equation 12 is the most critical challenge for tracing gradient descent methods in the generative AI era. Though several studies (Pruthi et al., 2020; Lin et al., 2024b) propose various techniques to reduce the computational complexity and achieve high usability for traditional machine learning models, the core requirement to instantiate and cache gradients for **every** training sample remains. This creates critical limitations when applying offline algorithms on generative AI models, which have exponentially growing parameter sizes and training datasets. Wang et al. (2025a) proposes *ghost dot-product*, a technique that calculates the *exact* inner product of a pair of data samples' gradients without instantiating them. The technique could also be leveraged for offline algorithms as well and serves as the current best practice.

Another challenge is the underperformance stemming from errors introduced by inherent assumptions and approximations. Although different resolutions (Xie et al., 2024a; Yeh et al., 2022) aim to improve gradient tracing performance, the deep, fundamental roots of these algorithms and their impact on downstream application performance require further investigation.

Offline tracing algorithms. Offline tracing algorithms are performed during model training. Since computing Equation 12 across every training iteration is computationally expensive, several techniques have been proposed to reduce the cost.

`TracIn-CP`. A straightforward solution is to only select a few checkpoints during the training process to approximate Equation 12. Pruthi et al. (2020) propose to further approximate the accumulation of these inner products using C **fixed** model checkpoints $\{\theta_{t_c}\}_{c=1}^{C}$ that are **sparsely saved** across the whole training process, where $C \ll T$. This leads to the following practical formulation of `TracIn-CP` :

$$\texttt{TracIn-CP}(z, z_{\text{test}}) \triangleq \sum_{c=1}^{C} \eta_{t_c} \langle \nabla_\theta f(z_{\text{test}}; \theta_{t_c}), \nabla_\theta \ell(z; \theta_{t_c}) \rangle, \tag{13}$$

where η_{t_c} is the learning rate at t_c. `Grad-Dot` is a special case when we extend `TracIn-CP` to use only one checkpoint, i.e., $C = 1$. It coincidences with the similarity in parameterized family defined in Charpiat et al. (2019).

Gradient vector compression. Another group of popular techniques are various ways of gradient vector compression, such as *layer cherry picking* and *random projection*. Assuming deep neural networks are usually overparameterized, Pruthi et al. (2020) propose to cherry-pick certain layers (e.g., only the last layer) of a deep neural network for gradient calculation to reduce the dimensionality of gradients. *Random projection* maps the parameter space to a lower-dimension subspace. Both of these techniques are also widely used in Influence Function (Section 2.2.2), thus we will not describe them in detail here.

Mitigating attribution underperformance. Previous efficient techniques applied to Equation 12, including checkpoint selection and gradient vector compression, may degrade the performance of offline algorithms across applications. To address this, recent studies identify and try to mitigate some specific failure modes. Yeh et al. (2022) discovers the *cancellation effect*, where last-layer weights make the attribution result of each sample have a large magnitude that contradicts each other. The authors propose picking the first layer (e.g., the embedding layer of a language model) for the tracing attribution algorithm. Separately, Xie et al. (2024a) finds that in diffusion models (Ho et al., 2020b), training examples with large diffusion timesteps (closer to noise) exhibit higher loss gradient norms, skewing the attribution results. They propose normalizing both training and test gradient terms by their respective norms.

The ease of implementation positions approaches centered around `TracIn-CP` as a go-to method for various applications (Akyürek et al., 2022). Per-example gradients are calculated (and optionally cached) for each sample at a maximum available batch size $|\mathcal{B}|$ constrained by the memory limit M_{limit}. Nevertheless, the per-sample gradient

instantiation is memory-consuming for large generative AI, which limits the maximum available batch size to $\frac{M_{\text{limit}}}{|\mathcal{B}|p}$, where p is the parameter size.

Online tracing algorithms. Online tracing algorithms aim to estimate the data influence during the model training process. By leveraging available training dynamics (e.g., backpropagation information), Wang et al. (2025a) enables efficient computation of gradient inner products between data sample pairs.

Ghost dot-product. For a typical neural net layer without parameter sharing, the per-sample gradient w.r.t. the parameters at this layer can be computed using the input to the layer and the gradient of the loss w.r.t. the output, both of which are available during backpropagation in batch. Furthermore, the inner product of two large gradients can be computed in a layer-by-layer fashion, avoiding the need to instantiate large gradient vectors. The formula for a linear layer can be presented as

$$\frac{\partial \ell^{(1)}}{\partial \mathbf{W}} \odot \frac{\partial \ell^{(2)}}{\partial \mathbf{W}} = \left(\mathbf{a}^{(1)} \otimes \frac{\partial \ell^{(1)}}{\partial \mathbf{s}^{(1)}}\right) \odot \left(\mathbf{a}^{(2)} \otimes \frac{\partial \ell^{(2)}}{\partial \mathbf{s}^{(2)}}\right) = \left((\mathbf{a}^{(1)})^\top \mathbf{a}^{(2)}\right) \left(\left(\frac{\partial \ell^{(1)}}{\partial \mathbf{s}^{(1)}}\right)^\top \left(\frac{\partial \ell^{(2)}}{\partial \mathbf{s}^{(2)}}\right)\right), \tag{14}$$

where $\ell^{(1)}$ and $\ell^{(2)}$ are the losses of the pair of data samples (typically a training sample and a test sample), $\mathbf{W}$ is the weight of the linear layer, $\mathbf{a}^{(1)}$ and $\mathbf{a}^{(2)}$ are the input of the linear layer, $\mathbf{s}^{(1)}$ and $\mathbf{s}^{(2)}$ are the pre-activate output of the linear layer, $\odot$ stands for the inner product and $\otimes$ stands for the outer product.

For online algorithms, $\mathbf{a}^{(1)}$, $\mathbf{a}^{(2)}$, $\frac{\partial \ell^{(1)}}{\partial \mathbf{s}^{(1)}}$ and $\frac{\partial \ell^{(2)}}{\partial \mathbf{s}^{(2)}}$ are ready during back-propagation, so that the overhead caused by ghost dot-product algorithm is relatively small. Nevertheless, the test samples need to be prepared **before** model training for online algorithms, which may be invalid for some use cases such as attributing a generated sample (Deng et al., 2024b). Ghost dot-product can also be applied to offline algorithms and intermediate checkpoints as well, which significantly reduces the space complexity. Detailed analyses are stated in the next paragraph.

Time and space complexity. We tabulate the time and space complexity of the methods in Table 7.

For complexity analysis, we split the complexity analysis to "One-time" and "Amortized". "One-time" stands for the complexity of calculating the data attribution result for one test samples on the full training dataset. "Amortized" stands for the amortized complexity of attributing continually incoming test samples and cache the gradient information if applicable.

We denote the size of training set as n, the parameter size as p, the number of training iteration as T, the number of checkpoints as C, the reduced parameter size (e.g., through random projection) as $\hat{p}$, and number of layer as L. In most cases $C \ll T$ and $L \ll p$. We also assume one forward pass of the target model costs F_θ, and one backward pass approximately costs two forward passes.

Table 7: Complexity analysis of `TracIn` and its variants. The complexity analysis here assumes non-sequential data and the attribution for the whole training process.

Methods	One-time		Amortized	
	Time	**Space**	**Time**	**Space**
`TracIn`	$O(nT(F_\theta + p))$	$O(p)$	$O(nTp)$	$O(nTp)$
`TracIn-CP`	$O(nC(F_\theta + p))$	$O(p)$	$O(nCp)$	$O(nCp)$
`TracIn-CP` **+ Gradient vector compression**	$O(nC(F_\theta + \hat{p}))$	$O(p)$	$O(nC\hat{p})$	$O(nC\hat{p})$
`TracIn-CP` **+ Ghost dot-product**	$O(nC(F_\theta + \sqrt{pL}))$	$O(\sqrt{pL})$	$O(nC(\sqrt{pL}))$	$O(nC\sqrt{pL})$

1. `TracIn` and `TracIn-CP` perform Equation 12 and 13 by directly instantiating per-sample gradients and caching them for multiple test samples. The only difference is the constant scaler C and T.
2. Gradient vector compression perform gradient compression right before the caching. Notably, this won't change the space complexity when calculating the gradient since mapping the gradient to a lower dimension happens after the instantiation.
3. Ghost dot-product could avoid explicitly instantiating per-sample gradient and reduce the space cost significantly, which is critical for large generative models. For each layer, caching $\mathbf{a}$ and $\frac{\partial l}{\partial \mathbf{s}}$ for one training samples takes $O(d_{in} + d_{out})$ space, which is proportional to the **square root of the original cost** of explicitly caching the per-sample gradient ($O(d_{in} \times d_{out})$).

Relationship with influence function. While both `TracIn` and Influence Functions (IF) are commonly used to quantify the *influence* of a training sample on a model's prediction for a test sample and both rely on relatively similar closed-form expressions. They differ in how they define *influence*. `TracIn` estimates the total change in the model's output during training process caused by a training sample (Equation 11). In contrast, IF approximates the change in the final model parameters and corresponding output when a training sample is removed from the training dataset (Equation 2). Another difference is that IF is agnostic to the optimizer used in the training process, while `TracIn` is strongly dependent on the optimizer. Stochastic gradient descent(SGD) is assumed to be the optimizer in the derivation of Equation 12, while it's not commonly used for most generative AI applications.

2.4.2 Trajectory-Specific LOO

The second type of dynamic method is trajectory-specific LOO (`TSLOO`), which was first proposed by Hara et al. (2019) in the name of SGD-influence. Like influence functions, `TSLOO` quantifies samples influence by analyzing the counterfactual scenario of LOO. However, unlike influence functions that only consider the counterfactual at the *final* model checkpoint, `TSLOO` examines the counterfactual at *all* intermediate checkpoints. In particular, it uses a first-order approximation to establish a recursive relationship between the LOO influences at consecutive iterations, thereby propagating a sample's influence from its initial appearance to the final model. Importantly, this approach does not assume that the loss function is convex or that the model is trained to convergence, both of which are key assumptions in deriving influence functions, thereby offering broader applicability[13]. Moreover, as we will see shortly, `TSLOO` eliminates the need to compute the inverse-Hessian-Vector Product (iHVP), which is the most computationally expensive component in influence functions. Instead, it requires only the computation of Hessian-vector products, which can be performed much more efficiently.

Notation. We denote the loss function as $\ell(z;\theta)$, where z represents a training sample and θ denotes the model parameters. We assume ℓ is twice-differentiable. The model is trained using SGD for T iterations. At iteration t, the model parameters are represented by $\theta^{[t]}$, while $\theta^{[t]}_{-j}$ denotes the model parameters in the counterfactual scenario where sample j is removed. The mini-batch used at this iteration is S_t, and the Hessian over the batch is given by $H^{[t]} = \frac{1}{|S_t|}\sum_{i\in S_t}\nabla^2_\theta \ell(z_i;\theta^{[t]})$. Finally, we define u as a query vector, which is typically chosen as the gradient of the loss of a test sample z_{test} at the final iteration, i.e., $u = \nabla\ell(z_{\text{test}};\theta^{[T]})$.

Original formulation of `TSLOO`. We review the core methodologies of `TSLOO` established by Hara et al. (2019), including both the derivation of its formula and an efficient implementation. We will also briefly discuss `HYDRA`, a closely-related method proposed by Chen et al. (2021).

Derivation of `TSLOO`. The high level idea of `TSLOO` is that, in order to obtain the LOO at the final model checkpoint, i.e., $\theta^{[T]}_{-j} - \theta^{[T]}$, it suffices to build a recursive relationship between the LOO at two consecutive iterations, $\theta^{[t]}_{-j} - \theta^{[t]}$ and $\theta^{[t-1]}_{-j} - \theta^{[t-1]}$.

We start with a simple case where the model is trained for only one epoch. Denote $\pi(j)$ as the timestamp where sample z_j is used in training, i.e. $z_j \in S_{\pi(j)}$. For $t > \pi(j)+1$, we have

$$\theta^{[t]}_{-j} - \theta^{[t]} = (\theta^{[t-1]}_{-j} - \theta^{[t-1]}) - \frac{\eta_{t-1}}{|S_{t-1}|}\sum_{i\in S_{t-1}}(\nabla_\theta \ell(z_i;\theta^{[t-1]}_{-j}) - \nabla_\theta \ell(z_i;\theta^{[t-1]})).$$

Using first-order expansion, we have the following approximation:

$$\frac{1}{|S_{t-1}|}\sum_{i\in S_{t-1}}(\nabla_\theta \ell(z_i;\theta^{[t-1]}_{-j}) - \nabla_\theta \ell(z_i;\theta^{[t-1]})) \approx H^{[t-1]}(\theta^{[t-1]}_{-j} - \theta^{[t-1]}).$$

Combining the above equations yields the following recursive relationship:

$$\theta^{[t]}_{-j} - \theta^{[t]} \approx (I - \eta_{t-1}H^{[t-1]})(\theta^{[t-1]}_{-j} - \theta^{[t-1]}), \quad t > \pi(j)+1.$$

Denote $Z_t = I - \eta_t H^{[t]}$, and note that $\theta^{[\pi(j)+1]}_{-j} - \theta^{[\pi(j)+1]} = \frac{\eta_{\pi(j)}}{|S_{\pi(j)}|}\nabla_\theta \ell(z_j;\theta^{[\pi(j)]})$, we have the following approximation of the LOO at the final model:

$$\theta^{[T]}_{-j} - \theta^{[T]} \approx \frac{\eta_{\pi(j)}}{|S_{\pi(j)}|} Z_{T-1}Z_{T-2}\cdots Z_{\pi(j)+1}\nabla_\theta \ell(z_j;\theta^{[\pi(j)]}) =: \Delta\theta_{-j}. \tag{15}$$

[13] On the other hand, to date `TSLOO` supports only the SGD optimizer and does not accommodate alternatives like Adam (Kingma and Ba, 2014), which are essential for training language models.

Intuitively, this formula can be interpreted as follows: starting from the first appearance of a sample, propagate its influence (scaled gradient) to the final iteration using second-order information. In data attribution literature, this approximation in Equation 15 first appears in Hara et al. (2019). Notably, similar analysis and expression appear frequently in related domains, including continual learning and deep learning theory (Zou et al., 2021; Evron et al., 2022; Wu et al., 2022a, 2024a; Ding et al., 2024).

For multi-epoch SGD, Hara et al. (2019) propose to sum up the influence of a sample at different epochs. Specifically, suppose SGD is applied for K epochs, and $\pi_k(j)$ is the timestamp of sample j's appearance at k-th epoch. We can similarly define $\Delta\theta_{-j}$ as

$$\Delta\theta_{-j} = \sum_{k=1}^{K} \left(\prod_{s=1}^{T-\pi_k(j)-1} Z_{T-s} \right) \frac{\eta_{\pi_k(j)}}{|S_{\pi_k(j)}|} \nabla_\theta \ell(z_j; \theta^{[\pi_k(j)]}),$$

Implementation of TSLOO. Hara et al. (2019) propose a *backward* computation algorithm that computes the TSLOO scores for *all* training samples in one pass. The key observation is that if the relevant information $(S_t, \eta_t, \theta^{[t]})$ is stored during training, it can later be used during inference to compute the (scaled) per-sample gradient and then its dot product with $u^{[t]}$, which can be sequentially updated from the initial query vector u. More specifically, starting from $t = T-1$ and proceeding backward to $t = 1$, at each iteration the algorithm computes $\langle u, \frac{\eta_t}{|S_t|} \nabla_\theta \ell(z_j; \theta^{[t]}) \rangle$ for every $j \in S_t$ and adds it to the TSLOO score of sample j. Then, the query vector is updated as $u \leftarrow Z_t u$. Additionally, Hara et al. (2019) leverage off-the-shelf implementations of the Hessian-vector product to compute $Z_t u$, thereby bypassing the challenges associated with directly computing and storing the Hessian.

Connection to HYDRA. Chen et al. (2021) propose HYDRA , which is based on a similar principle to TSLOO. Instead of computing the influence at the final model checkpoint directly via *implicit differentiation*, both methods *unroll* the differentiation throughout the optimization trajectory by establishing a recursive relationship between consecutive iterations. The key difference is that HYDRA considers the derivative of the upweighted objective as in influence functions, while TSLOO directly deals with LOO. In addition, HYDRA omits the Hessian terms in the recursive relationship for the purpose of computational efficiency, which results in a less precise approximation of the ground truth LOO.

Scalable variations of TSLOO. While the vanilla TSLOO has been applied to GAN (Terashita et al., 2021), vanilla TSLOO methods face significant challenges when the number of model parameters scales up. Since vanilla TSLOO requires unrolling the training and leveraging information such as the Hessian, it posts two main challenges: high computation cost and large storage demand. In this section, we introduce two scalable variations (Wang et al., 2025b; Bae et al., 2024) of TSLOO. We will mainly focus on their design and considerations to scale the method.

DVEmb. Wang et al. (2025b) points out that query vector u is unknown at training time, while $\Delta\theta_{-j}$ only requires information along training. The authors then propose data value embedding, noted as $\texttt{DVEmb}^{(t_s)}(z_j)$, defined as

$$\texttt{DVEmb}^{(t_s)}(z_j) =: \eta_{t_s} \left[\prod_{k=t_s+1}^{T-1} (I - \eta_k H^{[k]}) \right] \nabla \ell_\theta(z_j; \theta^{[t_s]}).$$

Notably, when $t_s = \pi(j)$, the definition of $\texttt{DVEmb}^{(t_s)}(z_j)$ is the same as Equation 15. Authors further leverage the empirical FIM to approximate the Hessian (see Section 2.2)

$$H^{[k]} = \sum_{z \in \mathcal{S}_k} \nabla \ell_\theta(z, \theta^{[k]}) \nabla \ell_\theta(z; \theta^{[k]})^\top,$$

which gives a recursive relationship for the proposed $\texttt{DVEmb}^{(t_s)}(z_j)$

$$\texttt{DVEmb}^{(t_s)}(z^*) = \eta_{t_s} \nabla \ell_\theta(z^*; \theta^{[t_s]}) - \eta_{t_s} \mathbf{M}^{(t_s)} \nabla \ell_\theta(z^*; \theta^{[t_s]}),$$

where the matrix $\mathbf{M}^{(t_s)}$ is defined as:

$$\mathbf{M}^{(t_s)} =: \sum_{t=t_s+1}^{T-1} \left(\sum_{z \in \mathcal{S}_t} \left(\texttt{DVEmb}^{(t)}(z) \nabla \ell_\theta(z; \theta^{[t]})^\top \right) \right).$$

To compute it recursively:

$$\mathbf{M}^{(t_s-1)} = \mathbf{M}^{(t_s)} + \sum_{z \in \mathcal{S}_{t_s}} \texttt{DVEmb}^{(t_s)}(z) \nabla \ell_\theta(z; \theta^{[t_s]})^\top,$$

and for $t = T-1$, $\mathbf{M}^{T-1} = \mathbf{0}$. The above framework allows computing $\mathbf{M}$ and DVEmb starting from $t = T-1$ to $t = 0$.

The method requires $O(TBp)$ storage during training, where B is the batch size and p is number of model parameters. This can be impractical especially for large models. To further improve the storage efficiency, the authors leverage gradient decomposition, storing only decomposed components instead of full gradient vectors, as in the *ghost dot product* (Equation 14), reducing storage requirements to $O(TB\sqrt{p})$ for non-sequential data. The author further includes dimensional reduction for foundation models so that the storage becomes $O(TB\tilde{p})$ where $\tilde{p}$ is significantly less than p.

The paper also includes some other considerations for computational efficiency. The authors propose to calculate data value embedding per layer so that the total FLOPs is reduced from $O(BT\tilde{p}^2)$ to $O(BT\tilde{p}^2/L)$ where L is the number of layers. The authors also propose parallelized computation through checkpointing the models for fixed step intervals, which reduce the iteration numbers.

SOURCE. Bae et al. (2024) introduces an averaged notion of SGD-influence that averages across all possible training trajectories rather than focusing on a single specific trajectory. To efficiently approximate this averaged influence, they partition the training process into multiple segments (e.g., corresponding to different training phases) and approximate the Hessian and gradient computations within each segment.

The segment-based TSLOO can be presented as following:

$$\mathbb{E}\left[\frac{d\theta^{[T]}}{d\epsilon}\right] = -\mathbb{E}\left[\sum_{\ell=1}^{L}\left(\prod_{\ell'=L}^{\ell+1}\mathbf{S}_{\ell'}\right)\underbrace{\left(\sum_{k=T_{\ell-1}}^{T_\ell-1}\frac{\eta_k}{B}\delta_k Z_{k+1:T_\ell}\mathbf{g}_k\right)}_{r_\ell}\right] \tag{16}$$

$$\approx -\sum_{\ell=1}^{L}\prod_{\ell'=L}^{\ell+1}\mathbb{E}\left[\mathbf{S}_{\ell'}\right]\mathbb{E}\left[r_\ell\right], \tag{17}$$

where $Z_k = I - \eta_k H^{[k]}$ and $Z_{k:k'} = Z_{k'-1}\cdots Z_{k+1}Z_k$. $\mathbf{S}_{\ell'} = Z_{T_{\ell'-1}:T_{\ell'}}$ indicates the influence of segment ℓ', r_ℓ indicates the influence of the iterations in segment ℓ, $\mathbf{g}_k = \nabla_\theta \ell(z;\theta^{[k]})$.

The approximation of stable Hessian and gradient within one segment ℓ greatly helps to further simplify $\mathbb{E}[\mathbf{S}_\ell]$ and $\mathbb{E}[r_\ell]$ to be

$$\mathbb{E}[\mathbf{S}_\ell] = \mathbb{E}[Z_{T_{\ell-1}:T_\ell}] \approx \left(I - \bar{\eta}_\ell \bar{H}^{[\ell]}\right)^{K_\ell} \approx \exp(-\bar{\eta}_\ell K_\ell \bar{H}^{[\ell]}) =: \bar{\mathbf{S}}_\ell,$$

$$\mathbb{E}[\mathbf{r}_\ell] = \mathbb{E}\left[\sum_{k=T_{\ell-1}}^{T_\ell-1}\frac{\eta_k}{B}\delta_k Z_{k+1:T_\ell}\mathbf{g}_k\right] \approx \frac{1}{N}(I-(I-\bar{\eta}_\ell\bar{H}^{[\ell]})^{K_\ell})(\bar{H}^{[\ell]})^{-1}\bar{\mathbf{g}}_\ell \approx \frac{1}{N}\underbrace{(I-\exp(-\bar{\eta}_\ell K_\ell \bar{H}^{[\ell]})(\bar{H}^{[\ell]})^{-1}\bar{\mathbf{g}}_\ell)}_{=:\bar{r}_\ell}.$$

Then the whole procedure has the following close form

$$\mathbb{E}\left[\frac{d\theta_T}{d\epsilon}\right] \approx -\frac{1}{N}\sum_{\ell=1}^{L}\left(\prod_{\ell'=L}^{\ell+1}\bar{\mathbf{S}}_{\ell'}\right)\bar{r}_\ell,$$

Since the calculation of earlier segments requires information from later segments, SOURCE features backward calculation: for later segments, the approximated Hessian product will be passed to earlier segments to assist calculation. Furthermore, SOURCE also proposes several techniques to scale the method. Firstly, by checkpointing, we require fewer calculations of Hessian matrices. Secondly, in order to efficiently calculate the Hessian and its product, SOURCE leverages Eigenvalue-corrected Kronecker-Factored Approximate Curvature (EK-FAC) (Grosse et al., 2023) to assume the Hessian. The authors also present a variant of SOURCE, called Fast-Source, in which the parameters are averages in a segment rather than Hessian and gradient.

2.4.3 Key Takeaways and Future Directions

Training dynamic methods are grounded in two seminal studies: `TracIN` (Pruthi et al., 2020) and `SGD-Influence` (Hara et al., 2019). Tracing gradient descent methods are popular due to their ease of implementation and relatively low computational cost, making them an active area of research for adaptation to various generative AI models. However, their performance can be compromised by spurious factors and often lags behind other methods in quantitative evaluations. Trajectory-specific leave-one-out (`TSLOO`) methods avoid some of the restrictive assumptions inherent in influence functions, such as convex loss functions and convergence during training, making them better suited for generative AI settings. Nevertheless, their scalability to large models remains a significant computational challenge, limiting their practical applicability. A common future direction for training dynamic methods is enhancing compatibility with modern optimizers. Both tracing gradient descent and Trajectory-specific LOO methods are derived under the assumption that stochastic gradient descent (SGD) is used, whereas SGD is rarely the optimizer of choice for training generative AI models. Xia et al. (2024) have taken initial steps toward addressing this mismatch, but many open questions remain.

2.5 Category: Simulator

This category of methods represents a critical line of research that uses surrogate models (i.e., simulators) to estimate the counterfactual effect of model training. `Datamodels` (Ilyas et al., 2022) is a landmark work in this category, introducing a conceptual framework for analyzing model behavior using a surrogate model and proposing linear datamodel as a practical surrogate function. Other methods in this category have proposed alternative simulators with varying input-output structures (Guu et al., 2023; Chai et al., 2024) and simulator architectures (Yoon et al., 2020; Liu et al., 2025). This section will also highlight the close connection between `Linear Datamodel` and semivalues (e.g., Data Banzhaf (Wang and Jia, 2023a)) in *weighted marginal contribution* category as well as how different simulation models extend the conceptual framework of `Datamodels`.

2.5.1 `Datamodels`

Notation. Suppose we have training dataset $D = \{z_i\}_{i=1}^n$. We extend the notation of *target function* as $f(z_{\text{test}}; \hat{\theta}_S)$ indicating the model output f on test data sample z_{test} and the model $\hat{\theta}_S$ trained on a subset of training dataset $S \subseteq D$.

Datamodeling. Ilyas et al. (2022) propose a data attribution framework, datamodeling, to quantify the influence of training data on model outcomes through surrogate functions. Specifically, `Datamodels` for a test sample z_{test} is a parametric function $g_\beta : 2^D \to \mathbb{R}$ optimized to predict $f(z_{\text{test}}; \hat{\theta}_S)$ given a training subset $S \subseteq D$:

$$g_\beta : \{0,1\}^n \to \mathbb{R}, \quad \text{where} \quad \beta = \arg\min_\beta \mathbb{E}_{S\sim\mathcal{D}}^{(M)} \left[\mathcal{L}_{\text{sim}}\big(g_\beta(\mathbf{1}_S); f(z_{\text{test}}; \hat{\theta}_S)\big) \right].$$

Here, $\mathcal{D}$ defines a sampling distribution of the subset S, a fixed training set. $\mathbf{1}_S \in \{0,1\}^n$ is the characteristic vector of S, $\mathcal{L}_{\text{sim}}(\cdot;\cdot)$ is a loss function for simulator training, and $\mathbb{E}^{(M)}$ is an M-sample empirical estimate of the expectation.

`Linear Datamodel`. In practice, Ilyas et al. (2022) propose to use a linear function as the surrogate function g_β, i.e., $g_\beta(S) = \beta_{[1:]}^\top \mathbf{1}_S + \beta_0$ for some $\beta \in \mathbb{R}^{n+1}$. This is referred to as the `Linear Datamodel`. The additive assumption simplifies the analysis by assuming that the combined effect of data can be presented as the simple summation of independent contributions from each point. Once this assumption is in place, a linear datamodel can be obtained via solving the following LASSO optimization problem (Tibshirani, 1996):

$$\beta = \arg\min_{\beta\in\mathbb{R}^{n+1}} \frac{1}{M} \sum_{k=1}^{M} \Big(g_\beta(\mathbf{1}_{S_k}) - f(z_{\text{test}}; \hat{\theta}_{S_k}) \Big)^2 + \lambda \|\beta\|_1. \tag{18}$$

The subsets $S_1, \ldots, S_M$ are sampled from a uniform distribution over α-fraction subsets of the training set, i.e., $\mathcal{D} = \mathcal{U}(S \subseteq D : |S| = \alpha n)$, which is a simple design choice made by Ilyas et al. (2022). $\lambda \geq 0$ is the regularization coefficient. The sparse prior regularization helps prevent overfitting and reduces the number of training needed. Finally, β_i will be the attribution score for the i-th training sample in the `Linear Datamodel`.

Connection with weighted marginal contribution methods. It is worth noting that the formulation in equation 18 is similar to the sparse regression estimator proposed in AME (Lin et al., 2022). We will further discuss the connection

between `Linear Datamodel` and *weighted marginal contribution* category. In general, `Linear Datamodel` can be written as linear combinations of utility values, or target function values, with proper designed subset sampling distribution $\mathcal{D}$.

When the l_1 regularization term is removed, we can reformulate Equation 18 as a weighted least-squares problem with a subset sampling distribution $\mathcal{D}$:

$$\arg\min_{\beta} \sum_{s\in[k]} w_s \Big[(\beta^\top \mathbf{1}_{S_s} \; - \; f(z_{\text{test}}; \theta_{S_s}))^2 \Big]. \tag{19}$$

where k subset are sampled from distribution $\mathcal{D}$, $w_s \geq 0$ and $\sum_{s=1}^{k+1} w_s = 1$. For `Linear Datamodel` with uniform sampling distribution, the weights are defined as

$$w_s = \begin{cases} \frac{1}{\binom{n}{\alpha n}}, & \text{if } |S| = \alpha n, \\ 0, & \text{otherwise.} \end{cases}$$

Considering the stacked vector $\mathbf{1}_S$ in X, W as the diagnal matrix with each element w_s, $f(z_{\text{test}}; \theta) \in \mathbb{R}^n$ as an output vector for all S, the optimal solution for this weighted least-squares problem is $\beta^* = (X^\top W X)^{-1} X^\top W f(z_{\text{test}}; \theta)$ provided $X^\top W X$ is invertible. The optimal solution β^* of these linear systems satisfies several key properties:

1. Under certain conditions on w_s, β^* preserves the pairwise differences of certain probabilistic values, that is, it retains the relative order of its components (Li and Yu, 2024b).
2. By placing a suitable distribution over the subsets (i.e., specific choice of w_s), β^* can recover known probabilistic values. In particular, setting $w_S = \frac{1}{2^n}$ and transforming X into a matrix with elements ± 1, we can recover Data Banzhaf Wang and Jia (2023a) (detailed in Appendix B). More related discussion can be found in (Li and Yu, 2024b).
3. For `Linear Datamodel`, the derived data values satisfies the *linearity* and *symmetry* axioms. Since β^* is derived via linear transformations of $f(z_{\text{test}}; \theta)$, linearity holds naturally. Symmetry follows because the weights w_s depend only on the subset size. If two data points contribute identically to $f(z_{\text{test}}; \theta)$ across all subsets, $X^\top W X$ and $X^\top W f(z_{\text{test}}; \theta)$ will remain symmetric w.r.t i and j, leading to $\beta_i^* = \beta_j^*$.

Ilyas et al. (2022) also discusses the connection of `Linear Datamodel` and empirical influence estimation proposed by Feldman and Zhang (2020). Specifically, taking $(f(z_{\text{test}}; \theta_S)$ as the correctness of predicting test sample z_{test}, the empirical influence estimation of i-th training data point on the test sample z can be approximated by:

$$\mathbb{E}_{S\sim\mathcal{D}}(f(z; \theta_{S\cup\{z_i\}})) - \mathbb{E}_{S\sim\mathcal{D}}(f(z; \theta_S)).$$

Ilyas et al. (2022) shows that with $\alpha = 0.5$ and transforming X into a matrix with elements ± 1, the minimizer of original linear regression in equation 19 can cast the estimated empirical influence in the infinite-sample case.

Challenges. There are two notable limitations to this approach of solving linear datamodel. First, the computational cost remains significant since, in equation 18, obtaining the ground truth labels requires training models on M subsets, and the dimension of $\boldsymbol{\theta}$ (i.e., the number of samples) is large. To resolve this problem, some gradient based works (Park et al., 2023) are proposed with promising results on Linear Datamodeling Scores (LDS), an evaluation metric proposed under the concept of `Linear Datamodel`. The method is introduced in Section 2.2.4 as its close relationship with influence function.

Second, the linear approximation of g_β may introduce *misspecification* errors, as it oversimplifies the potentially complex relationships in the data. These challenges highlight the need for further advancements to improve computational efficiency and reduce approximation errors.

2.5.2 Other Simulators

Other than Ilyas et al. (2022), different simulator-based data attribution methods are proposed to have different inputs, outputs and architectures in different studies. Two popular paradigms include 1) predicting the counterfactual target function (Yoon et al., 2020; Liu et al., 2025); 2) predicting the target function trajectory (Guu et al., 2023; Chai et al., 2024) are designed to resolve shortcomings in gradient-based data attribution methods.

Predict the counterfactual target function output. A group of methods that directly simulate the counterfactual target function output follow a simple idea that a map between training subset and target function output can be learned

by a neural network. The group of methods is closely related to `Linear Datamodel` (Ilyas et al., 2022), which can be seen as taken a linear model to predict the counterfactual target function.

DVRL (Data Valuation using Reinforcement Learning) (Yoon et al., 2020) leverages reinforcement learning to train a data value estimator (DVE). The method consists of two components, the target task predictor model and DVE that valuates each data example. On each training step, a target task predictor model is trained by minimizing the loss on the training set, weighted by the data valuation learned by DVE model. After that, the DVE model is optimized on a small validation set to minimize the loss through adjusting the data weight. The DVE model is trained with RL method policy gradient, receiving the reward signal from the validation loss and its moving average.

RegMix (Liu et al., 2025) is another work that leverages multiple retraining on subset to collect training data for the simulator to directly predict counterfactual model performance. The major difference between Liu et al. (2025) and Ilyas et al. (2022) is that RegMix take the data mixture percentage as simulator input rather than the sample-level information.

CLIMB (Diao et al., 2025) automatically clusters large datasets in semantic space and uses a small proxy model and a predictor to iteratively optimize the data mixture in the pre-training setting of LLMs.

Predict the training trajectory. Another group of method predict the training trajectory in each step, such as simfluence (Guu et al., 2023) and GPTfluence (Chai et al., 2024).

Simfluence (Guu et al., 2023) adopt a linear first-order Markov process to model the test loss throughout the training trajectory. The model takes in the curriculum, $\mathcal{C}_t$, which is the order of the training batches on time step t fed in, and the target function output before training step t begins and output predicted test loss after this training step. Here we simplify the target function to be the test loss on a test data point $\ell(z_{\text{test}}; \theta)$.

Specifically, the simulator is modeled as:

$$\ell(z_{\text{test}}; \theta_t) = \alpha(\mathcal{C}_t)\ell(z_{\text{test}}; \theta_{t-1}) + \beta(\mathcal{C}_t),$$

where $\alpha(\mathcal{C}_t) = \sum_{i \in \mathcal{C}_t} A_i$, $\beta(\mathcal{C}_t) = \sum_{i \in \mathcal{C}_t} B_i$, and $A_i, B_i \in \mathcal{R}^n$. The reduction in loss is composed by a relative reduction $\alpha(\mathcal{C}_t)$ and an absolute reduction $\beta(\mathcal{C}_t)$.

Similar to Simfluence (Guu et al., 2023), GPTfluence (Chai et al., 2024) propose another simulator model to predict the training trajectory. There are two major differences between Simfluence and GPTfluence. First, instead of a first order Markov process, the authors use an n-th order Markov process to model the training trajectory, which introduces more parameters but allows for modeling longer dependencies. Specifically, the simulator is modeled as:

$$\ell(z_{\text{test}}; \theta_t) = \sum_{j=1}^{n} \alpha_j(\mathcal{C}_t)\ell(z_{\text{test}}; \theta_{t-j}) + \beta(\mathcal{C}_t).$$

Note that the process is of n-order, meaning that the latest target function output is related to the previous n steps' target function output. Moreover, GPTfluence also proposed to a pre-trained encoder such as BERT and GPT to preserve the information within the data examples as well as support unseen training data in $\mathcal{C}_t$. The coefficients of the Markov process $\alpha(c_t)$ and $\beta(c_t)$ is then obtained by taking the inner product of these embeddings after some linear layers.

2.5.3 Key Takeaways and Future Directions

Training a simulator to directly predict attribution results represents a conceptual framework aimed at learning the relationship between training examples and model behavior. Among studies in this direction, Datamodels stands out as the most popular one. The primary challenge for this category of methods lies in the need to generate training data for the simulator—typically through model retraining. This requirement severely limits the scalability of simulator-based approaches to generative models. Although predicting the training trajectory has been proposed as a way to reduce retraining costs, it introduces another practical concern: error accumulation during trajectory prediction, which can significantly impact the reliability of the results.

2.6 Category: Other Approaches

There are also some methods that could not be grouped to previous categories. In this section, we collect the untouched methods and identify two rough groups: unlearning-based methods and similarity-based methods.

2.6.1 Unlearning-based Methods

A group of methods follow a general procedure that unlearn certain data points from the fully trained model and directly examine the counterfactual prediction through the unlearned model. Different methods are proposed to unlearn one or multiple data points, which can be broadly categorized into "*unlearning training samples*" and "*unlearning test samples*". Notably, most studies in this category focus on data attribution for diffusion models, likely due to the ambiguous definition and intriguing discover of the target function of diffusion model (Zheng et al., 2023; Lin et al., 2025) in gradient-based data attribution methods.

Unlearning training samples. A typical path is to retrain or fine-tuning on different subsets of training data (Lu et al., 2025; Dai and Gifford, 2023) as unlearning. The resulting unlearned models can then be used to estimate the marginal impact of individual training entities.

Dai and Gifford (2023) propose dividing the entire training set into n overlapping subsets, where n diffusion models are individually trained on these subsets. During the generation process, images are generated using an ensemble of these diffusion models. To unlearn a specific training example, all models that have seen the example are removed from the generation process. Similarly, Lu et al. (2025) fine-tune a fully trained model on different contributor-specific data subsets and compute the Shapley value for each contributor.

The path is closely connected to *weighted marginal contribution* and share some common challenges, where high computational cost of retraining and fine-tuning is the most critical one. To mitigate fine-tuning costs, Lu et al. (2025) first prune the model to approximate the fully trained version before fine-tuning.

Unlearning test samples. Another way is leveraging *the mirrored influence hypothesis* (Ko et al., 2024), which suggests a strong correlation between train-to-test and test-to-train influence. This insight enables the estimation of a training sample's marginal impact by unlearning the corresponding test sample (Wang et al., 2024h).

Similar to influence functions, Wang et al. (2024h) propose an attribution method for diffusion models. The key quantity of interest is the change in the loss of a training sample before and after unlearning a specific generated sample. Since retraining is computationally prohibitive, Wang et al. (2024h) employ unlearning techniques to approximate the removal effect. Specifically, they minimize the elastic weight consolidation (EWC) loss using multiple iterations of Newton update to mitigate the influence of the generated example.

$$\theta \leftarrow \theta + \frac{\alpha}{N} F^{-1} \nabla \ell(\hat{z}; \theta),$$
$$\tau(\hat{z}, z) = \ell(z; \theta_{-\hat{z}}) - \ell(z; \theta_0),$$

where F is the Fisher Information Matrix, ℓ is the training loss, $\hat{z}$ is the generated sample and z is a training sample.

2.6.2 Similarity-based Methods

Similarity-based data attribution methods aim to estimate the influence of data through a defined notion of similarity, following the intuition that influential training data should resemble test data. Based on this principle, several studies have proposed *model-agnostic* approaches for data attribution. However, these model-agnostic methods do not strictly fall within the scope of this survey, as our definition of data attribution in Section 2 focuses on model behavior. Therefore, we include only a few representative studies to illustrate research efforts in this direction.

Just et al. (2023) propose using multi-class Wasserstein distance to quantify the similarity between two datasets. They frame the data valuation problem as assessing how changes in the distribution of training data affect the similarity between the training and validation sets. Each training sample's contribution is then determined by computing the gradient of this distance with respect to perturbations in probability mass. Compared to LAVA, Yang et al. (2025) adopt a discrete similarity-based approach. Intuitively, they compute the value of a training sample as its calibrated distance to the entire validation set. Representer Point Selection (RPS) (Yeh et al., 2018) also employs weighted feature similarity to represent the importance of the training samples. Specifically, leveraging the representer theorem, it uses pre-activate input of model's last linear classification layer as feature and the resistance for training example feature towards minimizing the norm of the last layer's parameters as weight.

2.6.3 Key Takeaways and Future Directions

Machine unlearning is closely related to data attribution in terms of methodology, and thus faces similar challenges, such as the high computational cost of retraining and the non-convexity issues in gradient-based unlearning. The *mirrored influence hypothesis* offers a promising direction for reducing computational cost, particularly in scenarios with a small number of test samples and an extremely large training dataset. Similarity-based methods are generally model-agnostic; although they do not strictly align with the definition of data attribution, they are relatively easy to implement and can perform well in certain downstream tasks, such as mislabeled data detection.

3 Applications

In this section, we explore the applications of data attribution, focusing on generative AI settings. We categorize the applications into four primary areas: *data selection*, *fact tracing and concept probing*, *adversarial attack and defense*, and finally *data economy*. In addition, we discuss a few emerging but less common applications in Section 3.5.

For each application, we begin by defining the problem setup and objectives, emphasizing how data attribution principles can help address its specific challenges. We then review and organize relevant studies that employ attribution-based methods to tackle these challenges. In applications where literature in a generative AI context is still limited—given the recent emergence of attribution methods for generative models—we focus on conventional machine learning approaches instead, including them with the expectation that their insights can be transferred to generative AI settings. Finally, we conclude with a comprehensive synthesis of the current landscape and future directions. This closing section summarizes existing literature, reflects on unique difficulties and open challenges, considers practical aspects of applying attribution methods, and offers recommendations for future research and development.

3.1 Data Selection

Modern generative AI models are typically trained on extensive datasets to improve performance. However, not all data in these large corpora contribute positively to training outcomes: some examples may be irrelevant, mislabeled (Brodley and Friedl, 1999), redundant (Li et al., 2023), or even harmful (Gong et al., 2023). *Data selection* focuses on identifying and retaining the most useful data while filtering out less relevant portions (often referred to as *data cleaning*), with the aim of improving model accuracy and increasing training efficiency by reducing dataset size.

3.1.1 Problem Settings

Data selection aims to construct a curated training dataset D from a larger raw data pool D_{raw}, such that the model $M_D = \mathcal{A}(D)$ trained by algorithm $\mathcal{A}$ on D achieves desirable properties, e.g., low evaluation loss, better generalization, or safety guarantees. Formally, this corresponds to minimizing an objective function f_{obj} over possible datasets: $D^* = \arg\min_D f_{\text{obj}}(D)$, where D is drawn from a feasible subset of D_{raw} (Albalak et al., 2024). Notably, the selected dataset D need not be a strict subset; some approaches allow for weighted sampling or duplication, meaning D may contain repetitions or weightings of examples from D_{raw}. In general, the combinatorial nature of the search space renders exact optimization intractable.

To address this, a variety of approximation methods have been proposed. Recent work, in particular, has leveraged data attribution techniques, which estimate the *counterfactual* influence of individual training examples (or groups thereof) on a specified model behavior. These influence scores serve as tractable surrogates for the objective function and can guide data selection through computationally efficient heuristics. Thus, data attribution offers a scalable and principled approach to approximating the intractable selection problem.

Such approaches have been explored across a wide range of generative AI applications, varying in models and their training setup (e.g., LLM pre-training/fine-tuning, diffusion models), selection granularity (e.g., per-example, per-domain), and target objectives (e.g., model performance, safety). Before discussing the instantiations across these different settings, we first present a general algorithmic abstraction of data-attribution-based selection below.

General data-attribution-based data selection pipeline. To understand the core design principles of data-attribution-based selection, we begin by identifying the main computational bottlenecks in directly solving $\arg\min_D f_{\text{obj}}(D)$. First, the combinatorial explosion of candidate datasets D renders exhaustive search intractable. Second, evaluating $f_{\text{obj}}(D)$ even once typically requires full training and evaluation, which is prohibitively expensive in large-scale generative models. In practice, only a limited number of such "oracle" evaluations of f_{obj} are computationally feasible.

Data attribution techniques address both challenges by replacing exhaustive evaluation with efficient *counterfactual prediction*, followed by a computationally tractable *selection strategy*. Conceptually, counterfactual prediction provides a localized estimate of each training unit's *influence*, and the selection strategy operates on these scores to curate the final dataset D. We note that this general framework has been widely explored in recent data selection work, particularly in the context of large language models (Albalak et al., 2024).

In what follows, we focus on the distinctive features of data-attribution-based methods for both counterfactual prediction and the selection strategy. The former offers a more principled and interpretable approach to data selection compared to heuristic or proxy-based alternatives. The latter, while more general and not unique to attribution-based pipelines, remains integral to the overall effectiveness and trade-offs of the framework.

Counterfactual prediction. Instead of directly evaluating the objective for every possible dataset D, counterfactual prediction considers a localized, fine-grained *target function* $f(S)$, commonly defined as

$$f(S) \coloneqq f_{\text{obj}}(D_{\text{raw}} \setminus S),$$

where S is a subset of the training data, such as a single point, a group of examples, or an entire domain. This function captures the sensitivity of f_{obj} to small perturbations of the dataset and reflects the counterfactual change in the objective when S is removed. Specifically, the corresponding counterfactual change is referred to as the *influence* of S, defined as

$$\tau(S) \coloneqq \Delta f(S) = f(S) - f(\varnothing) = f_{\text{obj}}(D_{\text{raw}} \setminus S) - f_{\text{obj}}(D_{\text{raw}}),$$

which quantifies how much the objective would degrade (or improve) if S were removed. This measures the contribution of training units to the property encoded in the objective. For example, when the goal is to improve model performance, one may define the objective as $f_{\text{obj}}(D) = \ell(D_{\text{eval}}; M_D)$, where ℓ is a loss function evaluated on a fixed evaluation set D_{eval}. A natural choice of target function is then $f(S; D_{\text{eval}}) \coloneqq \ell(D_{\text{eval}}; M_{D_{\text{raw}} \setminus S})$, and the corresponding influence becomes $\tau(S) = \ell(D_{\text{eval}}; M_{D_{\text{raw}} \setminus S}) - \ell(D_{\text{eval}}; M_{D_{\text{raw}}})$, capturing the marginal impact of S on evaluation performance.

Crucially, various *data attribution* algorithms exist to approximate $\tau(S)$ without explicitly retraining models for each possible subset S, which enables scalable, localized assessment of individual data units' impact on the objective. We note that the specific choice of data attribution algorithm depends on several factors, including the model training setting (e.g., pre-training, fine-tuning), the form of the target function (e.g., evaluation loss, likelihood), and the granularity of attribution (e.g., individual points, groups, or domains). With a choice of data attribution method, the resulting estimated influence scores $\tau(S)$ then serve as surrogates to approximate the original dataset-level optimization objective in a computationally tractable manner.

Selection strategy. The final ingredient in the data-attribution-based data selection pipeline is to act on the estimated influence scores by deciding which data to retain, discard, or re-weight. This is governed by a *selection strategy*, which translates local influence estimates into global modifications of the dataset. Although not unique to attribution-based methods, the selection strategy plays a crucial role in determining how effectively we can utilize the localized estimates $\tau(S)$ to approximately maximize the original dataset-level objective $f_{\text{obj}}(D)$. As the influence scores are defined for every small training unit S (e.g., individual examples or groups), several widely used strategies arise naturally by making decisions based on these individual scores:

- *Threshold* selects data units whose influence exceeds (or falls below) a fixed or adaptive threshold. This avoids combinatorial search by assuming that influence scores provide a meaningful separation between helpful and harmful data.
- *Top-k* selection fixes the number of retained examples, selecting the k highest-ranked units by influence. This strategy assumes that influence magnitudes provide a globally consistent ordering of utility.
- *Importance weighting* retains all data but adjusts each unit's contribution according to its influence. This relaxes the binary nature of selection and assumes that marginal improvements can be linearly composed.

These strategies each reflect specific structural or heuristic approximations to the original dataset-level optimization, and entail trade-offs: threshold is simple but may be brittle under noisy influence estimates; top-k offers explicit control over dataset size but may discard complementary examples; importance weighting preserves more information but depends critically on the fidelity of influence estimation. Crucially, all of these approaches rely on a common modeling assumption: that each training unit's influence on the objective can be assessed independently of others. That is, we assume $\tau(S \cup S') = \tau(S) + \tau(S')$, or at least that τ is highly additive or positively correlated across units, so that making local decisions independently (i.e., selecting S into D or not) yields a near-optimal global outcome $f_{\text{obj}}(D)$. This additive or decomposable view of influence enables tractable approximation of the otherwise combinatorial selection problem. However, this assumption may not always hold: for instance, when the utility of one point depends on the presence of others (Hu et al., 2024a).

Lastly, we also highlight that selection strategies can be deployed in either *offline* or *online* settings. In the offline setting, data is selected prior to training based on influence scores computed using a fixed model, typically trained on a subset of the full dataset as a proxy. This setting offers computational efficiency and is well-suited to static selection. In contrast, online selection interleaves influence estimation and model training, often by periodically re-evaluating scores as the model evolves. This enables more adaptive selection that can account for shifting data utility during training, and also helps mitigate the additive limitation discussed above by dynamically updating scores to partially capture higher-order interactions. The two approaches can also be integrated: for example, by initializing with an offline selection and refining it online.

3.1.2 Landscape of Data-Attribution-Based Data Selection in Generative AI

In this section, we examine how concrete data selection algorithms instantiate key aspects of data attribution methods—counterfactual prediction and selection strategy—across diverse generative AI training settings, to achieve the specified objective. Specifically, in each setting, we structure the discussion as follows: we first identify the general objectives and key obstacles shared in this setting, and discuss how different data attribution algorithms and selection strategies are utilized to overcome them together with specific considerations that are unique to different works.

Data selection for LLM pre-training. In LLM pre-training, the goal is to acquire LLMs with "general knowledge" from the massive text corpora collected from various sources and domains. Due to the massive raw training dataset D_{raw}, the main bottleneck at this stage is the intense computational resources required for a pre-training run. Therefore, data selection becomes a natural technique to reduce the training cost by reducing the dataset size.

In the case of data-attribution-based selection methods, as pre-training focuses on acquiring "general knowledge," the objective $f_{\text{obj}}(D)$ is often the validation loss $\ell(D_{\text{eval}}; M_D)$, with the natural target function $f(S) = f_{\text{obj}}(D_{\text{raw}} \setminus S) = \ell(D_{\text{eval}}; M_{D_{\text{raw}} \setminus S})$ as discussed above (Zhang et al., 2025a; Wang et al., 2024e; Liu et al., 2025). In some special cases when the downstream tasks are known in advance, the evaluation set D_{eval} can be drawn specifically from the task domains (Wang et al., 2023c; Yu et al., 2024). On the other hand, Thakkar et al. (2023) approaches the problem with a complementary viewpoint by focusing on identifying noisy and outlier data. In such a scenario, *self-loss* $f(S) = \ell(S, M_{D_{\text{raw}} \setminus S})$ becomes a natural target function with its induced influence score being the so-called *self-influence* since it has been observed that self-influence is a strong indicator of outlier and noisy data (Bejan et al., 2023).[14]

Given a target function, various attribution methods can be applied to estimate counterfactual effects, which are then used in conjunction with a selection strategy to curate the final dataset. However, due to the high computational cost of estimating counterfactual influence scores—especially in large-scale generative models with massive raw training dataset—the main consideration in choosing a specific attribution method and selection strategy is minimizing additional overhead. Below, we summarize three common optimization strategies for achieving this goal:

- *Efficient influence approximation*: The most direct approach is to reduce the computational burden by accelerating the attribution step itself. A wide range of methods have been proposed to scale up standard attribution algorithms (Section 2), including utilizing proxy models, exploiting model architectures, or more aggressive approximations that trade off prediction fidelity for speed. We will expand on these techniques in later paragraphs.
- *Coarser granularity*: Most prior work adopts per-example attribution (i.e., $|S| = 1$), which requires estimating $|D_{\text{raw}}|$ separate influence scores. A natural way to reduce this cost is to increase the attribution granularity, e.g., performing attribution over groups or domains (i.e., $|S| > 1$), which reduces the total number of training units that must be evaluated and selected.
- *Reduced frequency of selection*: Another orthogonal strategy is to minimize the number of times attribution and selection are performed. Offline selection methods typically require only a single attribution pass before pre-training, whereas online methods continuously re-estimate influence during training. While online strategies may improve adaptivity, they incur substantially higher overhead due to repeated evaluation.

Recent works explore various *combinations* of these optimization strategies to balance computational efficiency with attribution fidelity. Table 8 summarizes representative methods applied to LLM pre-training, where bolded entries indicate those that explicitly incorporate one of the above strategies to reduce computational cost. Note that this is omitted in the **Attribution Method** column, as almost all works have a certain level of optimization to one of the standard attribution algorithms. In particular, we highlight several representative works that exemplify distinctive pipelines or introduce algorithmic innovations worth discussing from various angles.

[14] As an intuitive explanation is that, if removing a data point deteriorates the loss value on itself, then the data should be different enough so that the prediction on which could not be learned from the rest of the data, making it an outlier in most cases.

Table 8: Comparison of data-attribution-based methods for LLM pre-training.

References	Attribution Method	Target Function	Granularity	Selection Strategy
Thakkar et al. (2023)	`TracIn`	*Self-Loss*	Per-example	Re-weighting & Online
Wang et al. (2023c)	`Grad-Dot`	Validation Loss	Per-example	Top-k & **Offline**
Wang et al. (2024e)	(Higher-order) `Grad-Dot`	Validation Loss	Per-example	Top-k & Online
Yu et al. (2024)	BERT-based Datamodel	Validation Loss	Per-example	Top-k & Online
Engstrom et al. (2024)	`TRAK` on Proxy Model	Validation Loss	**Subset**	Top-k & **Offline**
Zhang et al. (2025a)	`IF`	Validation Loss	Per-example	Top-k & **Offline**
Liu et al. (2025)	Regression Datamodel	Validation Loss	**Domain**	Re-weighting & **Offline**

Firstly, a popular research direction has focused on improving the efficiency of gradient-based attribution methods by exploiting structural properties of gradients, often in the form of *Kronecker factorization*, to speed up influence computation. For example, Wang et al. (2024e) proposes the *ghost dot-product* for linear layers, which enables efficient influence estimation (Equation (14)). Similarly, Zhang et al. (2025a) leverages a Kronecker factorization structure tailored to attention layers, achieving comparable acceleration in transformer-based models. These algorithmic refinements lead to notable speedups compared to their original attribution counterparts.

Next, we revisit the distinction between online and offline selection strategies and how attribution methods adapt in each case. As discussed earlier, online selection has the conceptual advantage of using the current model checkpoint as the reference model for attribution. This makes it natural to apply *dynamic* attribution methods, which are designed to trace the behavior of a specific training run. For example, Thakkar et al. (2023) uses `TracIn` to compute self-influence scores for individual samples throughout training, enabling an adaptive re-weighting scheme that evolves over time. Along similar lines, Yu et al. (2024) trains a separate influence prediction model to approximate oracle scores in real time. This model is combined with a Gumbel-top-k sampling mechanism to introduce stochasticity into data selection, thereby balancing exploration and exploitation during pre-training.

In contrast, offline strategies face a conceptual challenge: since training has not yet started, no reference model $M_{D_{\text{raw}}}$ exists to support counterfactual prediction. A common workaround is to train small proxy models at low cost and use them for attribution. For instance, Liu et al. (2025) frames domain-level selection as a regression task, using small models trained on various data mixtures to estimate domain influence. Similarly, Engstrom et al. (2024) trains multiple GPT2-small variants on diverse subsets and applies `TRAK` to identify influential subsets for pre-training. An alternative approach is to assume access to a partially trained model: both Zhang et al. (2025a) and Wang et al. (2023c) consider a setting in which a pre-trained model already exists, and the goal is to refine it further. They hypothesize that influence estimates computed with respect to this intermediate model are sufficiently accurate to guide effective data selection.

Data selection for LLM fine-tuning. LLM fine-tuning focuses on improving generalization and robustness beyond a pre-trained foundational model by training on task-specific examples. Compared to pre-training, this setting typically involves domain-specific, high-quality *annotated* data and operates at a smaller scale, making it more controlled and selective. As a result, more computational resources can be allocated to the data selection process, enabling more precise and confident decisions to maximize data efficiency.

In the context of data-attribution-based selection for LLM fine-tuning, the choice of target functions is more diverse than in pre-training, as overall task performance is not always the sole objective. While validation loss on a general held-out evaluation set D_{eval} remains a common choice (Jiao et al., 2025; Chhabra et al., 2024; San Joaquin et al., 2024; Xia et al., 2024; Wang et al., 2024g; Cao et al., 2024b; Zhou et al., 2024), other target functions have been explored to reflect different priorities. For example, Zhao et al. (2024) considers federated fine-tuning, where private client data cannot be shared; accordingly, the target function is defined using a *public* validation set. He et al. (2024) focuses on safety-critical applications, where the validation set is curated to emphasize safety-sensitive examples. Han and Tsvetkov (2021), in contrast, targets robustness to spurious correlations: their objective is to minimize loss on a held-out set that reflects only true task-relevant features, thereby penalizing training examples that strongly influence spurious signals. To summarize, although validation loss remains the predominant choice of target function in LLM fine-tuning, the design and curation of the evaluation set can vary significantly depending on task-specific goals, such as safety, privacy, or robustness, leading to different downstream data selection behaviors.

With the target function defined, various attribution methods are applied to conduct the corresponding counterfactual predictions, followed by a selection strategy to curate the final dataset. Interestingly, in contrast to the optimization strategies commonly explored for LLM pre-training—such as *efficient influence approximation* and *coarser granularity*—

data selection for fine-tuning often takes a different direction, emphasizing *precision* over efficiency due to the smaller and more curated nature of D_{raw}. Two common design choices reflect this shift:

- *Precise influence estimation*: Unlike in pre-training, where attribution efficiency is a primary concern, fine-tuning literature places less emphasis on speed and instead adopts computationally expensive methods that align closely with task objectives. Classical influence functions, Shapley-value-based methods, and dynamic approaches are all explored in this setting.
- *Finest granularity*: Because fine-tuning datasets are typically small and costly to collect, *per-sample* attribution becomes a natural choice, as this granularity enables highly targeted and precise selection decisions to achieve data efficiency.

We remark that offline methods remain widely used: unlike in pre-training, the absence of a reference model is not a bottleneck in fine-tuning since the pre-trained model itself provides a strong initialization. Since the fine-tuned model is often assumed to remain close to its pre-trained counterpart, attribution scores computed *a priori* with respect to the pre-trained model tend to remain valid throughout the fine-tuning process. This makes offline, one-shot estimation of influence both practical and reliable in this regime. In summary, recent work often combines both of the above strategies to enable accurate data selection in data- and resource-constrained environments.

Table 9: Comparison of data-attribution-based methods for LLM fine-tuning.

References	Attribution Method	Target Function	Granularity	Selection Strategy
Han and Tsvetkov (2021)	`Grad-Dot`	**Validation Loss**	Per-example	Custom & Online
He et al. (2024)	`Grad-Dot`	**Validation Loss**	Per-example	Top-k & Offline
Xia et al. (2024)	`TSLOO`	Validation Loss	Per-example	Top-k & Offline
Wang et al. (2024g)	Shapley	Validation Loss	Per-example	Top-k & Offline
Cao et al. (2024b)	Regression Datamodel	Validation Loss	Per-example	Top-k & Offline
Zhou et al. (2024)	`IF`	Validation Loss	Per-example	Top-k & Offline
San Joaquin et al. (2024)	`IF`	Validation Loss	Per-example	Top-k & Offline
Zhao et al. (2024)	`IF`	**Validation Loss**	Per-example	Threshold & Offline
Chhabra et al. (2024)	`IF`	Validation Loss	Per-example	Custom & Offline
Jiao et al. (2025)	`IF`	Validation Loss	Per-example	Top-k & Online

Table 9 summarizes representative methods applied to LLM fine-tuning. Bolded entries in the **Target Function** column indicate cases where the validation loss is evaluated on a carefully curated evaluation set to reflect task-specific goals, as discussed earlier. Below, we highlight several notable works that illustrate distinctive algorithmic choices and design decisions in this regime.

Computation-wise, most of the referenced methods rely on relatively expensive attribution algorithms compared to those used in LLM pre-training. Among them, Wang et al. (2024g); Cao et al. (2024b) is particularly notable for employing weighted marginal contribution methods, which generally require multiple rounds of re-training. While computational cost is somewhat less prohibitive in the fine-tuning setting, repeated fine-tuning remains impractical at scale. To address this, Cao et al. (2024b) leverages proxy models techniques as we have seen in LLM pre-training; on the other hand, Wang et al. (2024g) approximates Shapley values using the empirical neural tangent kernel (NTK), which avoids explicit retraining by assuming the model remains close to its pre-trained initialization, which is a reasonable assumption in the fine-tuning regime (see Section 2.3).

Beyond attribution itself, the fine-tuning setting also permits more complex training-time integration of selection signals. For example, Han and Tsvetkov (2021) proposes an *online* and *differentiable* selection strategy by directly incorporating attribution scores into the training objective. The model is guided to prioritize samples aligned with a curated validation set, while penalizing those that reinforce spurious correlations. As the underlying attribution scores are estimated via Grad-Dot, this method requires computing higher-order gradients during training, making it computationally intense compared to more straightforward selection strategies. This broader theme of tailoring attribution pipelines to task-specific goals is also reflected in other designs. Chhabra et al. (2024), for instance, applies outlier detection algorithms directly on attribution scores rather than raw input features to identify and discard anomalous training examples.

Data selection for large vision-language models. Large vision-language models (LVLMs) have emerged to meet the growing demand for foundational models capable of processing multi-modal inputs, particularly combining visual data with text. A typical training paradigm, following the foundational work of Liu et al. (2023), adopts a *two-stage* process:

pre-training followed by *fine-tuning*. This setup assumes modality-specific backbone models, which typically consist of a pre-trained LLM for text and a pre-trained visual encoder. In the pre-training stage, LVLMs focus on aligning representations across modalities, while the fine-tuning stage adapts the model to specific downstream vision-language tasks, much like in LLM fine-tuning.

A (potentially surprising) key challenge in **both** stages is that the training data is typically large-scale and noisy, similar to LLM pre-training. This is especially true when synthetic vision-language datasets are created by pairing raw modality-specific data, where even modest base datasets (e.g., typical fine-tuning datasets) can lead to a combinatorial explosion in the number of examples. Consequently, data selection again emerges as a natural technique to reduce training cost by prioritizing high-utility samples.

Table 10: Comparison of data-attribution-based methods for LVLM pre-training *(Top)* and fine-tuning *(Bottom)*.

Reference	Attribution Method	Target Function	Granularity	Selection Strategy
Zhou et al. (2024)	`IF`	Validation Loss	Per-example	Top-k & Offline
Liu et al. (2024b)	`Grad-Dot`	Validation & Self-Loss	Per-example	Top-k & Offline

As LVLMs are still a relatively new research area, only a few works have explicitly explored data selection in this context. We summarize two representative studies in Table 10: Zhou et al. (2024), which focuses on pre-training, and Liu et al. (2024b), which addresses fine-tuning. Both adopt *validation loss* as a core target function. A distinguishing feature of Liu et al. (2024b) lies in its design of the target function: since the fine-tuning task typically spans multiple subtasks, each with potentially different characteristics, the authors augment the standard validation loss with a measure of *task difficulty*. Specifically, they compute the average *self-loss* of samples within each task as a proxy for task difficulty, and combine this with validation loss to guide more nuanced data selection across subtasks.

To mitigate the potentially high computational cost of counterfactual prediction, both works adopt techniques that echo those used in LLM pre-training—particularly *efficient influence approximation* and *reduced frequency of selection*. In terms of attribution algorithms, Zhou et al. (2024) introduces a highly optimized influence function based on the *hyperpower method*, specifically Schulz's iterative algorithm, while Liu et al. (2024b) employs Grad-Dot as a scalable approximation.

On the selection side, both studies implement a simple top-k selection strategy in an offline setting. Importantly, this choice is not just a pragmatic trade-off between accuracy and efficiency; it is also conceptually motivated. In the case of LVLM pre-training, where the goal is primarily to *align* features between backbone encoders (usually through training additional adapters by fixing the pre-trained backbone encoders' weights), and hence the model is not expected to evolve significantly during training. The same happens in the case of LVLM fine-tuning. As a result, attribution scores computed with respect to the initial model can remain valid throughout training, making offline selection a reasonable and effective approach.

Data selection in the pre-LLM era. In contrast to recent trends in large language models, earlier work on generative models focused on narrower NLP tasks such as machine translation with smaller model sizes. In these settings, data selection was primarily geared towards *data cleaning*, where identifying outliers and error-prone instances played a crucial role. Robustness was essential because these approaches typically involved end-to-end training of text-generation models. Moreover, the smaller scale of these tasks made repeated re-training feasible, thus justifying and facilitating offline selection.

Table 11: Comparison of data-attribution-based methods for LM training in the pre-LLM era.

Reference	Attribution Method	Target Function	Granularity	Selection Strategy
Lam et al. (2022)	Influence Function	Validation Loss	Per-example	Top-k & Offline
Bejan et al. (2023)	Influence Function and TracIn	Self-Loss	Per-example	Custom & Online
Li et al. (2024a)	Custom Heuristics	Likelihood Loss	Token-level	Custom & Offline

We summarize three representative works in Table 11, which collectively exhibit a diverse range of techniques and creative problem formulations. Among these, Bejan et al. (2023) and Li et al. (2024a) stand out for their distinctive, custom approaches. Bejan et al. (2023) investigates the role of self-influence in data selection and proposes an online, *learnable* selection mechanism that leverages automated curriculum learning; by integrating with dynamic attribution methods such as TracIn, their approach naturally supports online decision-making that adapts to evolving model behavior. On the other hand, Li et al. (2024a) takes a fine-grained approach by analyzing the distribution of non-target

tokens during text generation. They observe that a high overall loss for a token may arise either from inherently ambiguous next-token distributions or from actual mispredictions (or, equivalently, wrong labels). To differentiate between these cases, they compute an ℓ_2 error norm between the predicted distribution and the one-hot target, thereby considering all non-target tokens. The hypothesis is that training on tokens with excessively high ℓ_2 error (indicative of "wrong" label) leads to non-robust performance. Based on this insight, their custom selection strategy truncates tokens with error norms above a preset threshold by setting their loss to zero—effectively filtering out noisy or high-entropy training signals rather than discarding full examples.

Overall, these methods reflect the creative and varied approaches developed during the pre-LLM era, offering valuable insights into how data cleaning and selection strategies can enhance the robustness of generative language models.

3.1.3 Discussion

Data-attribution-based methods for data selection aim to directly estimate the counterfactual impact of modifying the training set with respect to a target function, offering actionable insights for dataset refinement. While the objective and constraints vary across application settings, several common challenges persist:

- **Additivity assumption**. As discussed earlier, common selection strategies such as top-k and thresholding implicitly assume that attribution scores are additive across training units. That is, the effect of a subset is approximated by summing per-example scores. However, this assumption is not theoretically justified for many popular attribution methods, including influence functions (Hu et al., 2024a) and data Shapley (Wang et al., 2024f). Accurately attributing the collective effects of data subsets remains computationally challenging. Online selection strategies offer a partial remedy by capturing marginal effects dynamically during training.
- **Dependence on a reference model**. A fundamental challenge in attribution-based selection is its post-hoc nature: attribution scores are computed with respect to a reference model, typically trained on the full raw dataset D_{raw}. This presents a dilemma—training on D_{raw} to enable attribution defeats the purpose of data selection in resource-constrained settings. Two common workarounds have been explored: one is using lightweight proxy models as stand-ins for the full model, and another is to adopt online attribution methods that estimate influence during training without requiring a fully trained model upfront.
- **Sensitivity and robustness**. Since data selection aims to minimize human intervention and operate autonomously, the reliability of attribution scores is paramount. If these scores are noisy or biased, there is no clear mechanism for validation or correction. Attribution methods are known to be sensitive to input perturbations (Koh and Liang, 2017) and vulnerable to adversarial manipulation (Wang et al., 2025c). Despite their growing use, robust and trustworthy attribution algorithms remain an underexplored research direction.

Despite these challenges, data-attribution-based methods offer several key advantages over alternative data selection approaches. Unlike classifier-based filtering (Raffel et al., 2020; Longpre et al., 2024), which often lacks transparency, or heuristic filtering (Gan et al., 2024; Albalak et al., 2024), which can introduce unintended biases or degrade performance, attribution-based selection provides a more principled and interpretable foundation for identifying impactful training data. For a broader overview of alternative data selection strategies, we refer readers to the survey by Albalak et al. (2024).

In summary, data-attribution-based methods for data selection offer a flexible and principled framework that can accommodate a wide range of desiderata in trustworthy machine learning, simply by adjusting the choice of target function. Compared to alternative approaches, attribution-based methods provide greater headroom for improvement, particularly when the benefits of heuristic-based selection begin to saturate (Wang et al., 2025a). As such, they can serve as a *complementary* component in broader data selection pipelines. A promising hybrid strategy involves using classical filtering techniques for coarse pruning, followed by data-attribution-based methods for more fine-grained selection and cleaning. This combination balances computational efficiency with selection precision, making it especially well-suited for large-scale generative AI applications.

We also note that while current work on data-attribution-based selection has primarily focused on LLMs, other generative settings, such as image generation via diffusion models, remain largely unexplored. Although recent efforts have advanced data attribution techniques for diffusion models (Lin et al., 2025; Xie et al., 2024a), their integration into data selection pipelines has yet to be studied. Extending attribution-based selection to vision generative models thus represents a promising direction for future research.

3.2 Fact Tracing and Concept Probing

Modern generative models have made remarkable progress in both text and image generation. However, their complexity and opaque generation processes pose significant challenges in understanding how the content is generated. A key

concern is whether their outputs are grounded in accurately captured prior concepts or derived from factual training data. In this context, *fact tracing* and *concept probing* have emerged as essential analytical tools for tackling these tasks.

3.2.1 Problem Settings

Fact tracing seeks to identify and verify the specific training data sources that contribute to a model's generated output, helping to establish the factual basis of its responses. For LLMs, an important implication of fact tracing is to resolve the *hallucination* phenomenon (Huang et al., 2025), where models generate misleading or entirely fabricated information, making it difficult for researchers and users to assess the reliability and authenticity of the generated content. To address this issue, researchers have started exploring techniques such as question-answering attribution (Bohnet et al., 2022), which focuses on benchmarking and building systems that trace model responses back to their source data. These efforts underscore the growing need for methods that enhance transparency and trust in LLMs' outputs.

In contrast, *concept probing* aims to disentangle and analyze the internal representations of abstract concepts within the model that stem from the training data. While the notion of "concept" is often vague and can vary depending on the specific context, there are growing efforts that focus on common use cases such as multi-lingual LLMs (Choenni et al., 2023, 2024b,a) and generative diffusion models (Brokman et al., 2024). For instance, concept probing for diffusion models is usually coined as *image tracing*, as it involves identifying training images with certain high-level concepts such as style, subject, or category that contribute to a generated image (Xie et al., 2024a).

By systematically applying these techniques, researchers can better interpret model behavior, assess reliability, and mitigate risks associated with misinformation.

General data-attribution-based fact tracing pipeline. Let $\mathcal{F}$ denote the set of factual knowledge or concepts of interest. In fact tracing or concept probing for generative models, the goal is to identify which portion of the training data help the model to acquire certain facts. A fact tracing or concept probing pipeline typically starts by observing that the model tends to prefer a given fact or concept over its negation, i.e. $p(t) > p(\neg t)$ for $t \in \mathcal{F}$, where t is a specific fact, $\neg t$ is its counterfactual, and $p(\cdot)$ denotes the probability of the model giving a certain prediction. In contrast to fact tracing, hallucination detection focuses on the set of logical negations $\neg\mathcal{F} = \{\neg t \mid t \in \mathcal{F}\}$, identifying instances where the model generates unsupported or fabricated content.

Once the target facts for attribution $\mathcal{F}$ is defined, data attribution methods can be applied to trace the model's behavior on $\mathcal{F}$ back to its training data. Specifically, these methods aim to answer: *How would the model's behavior on $\mathcal{F}$ change if certain training instances were removed, and how significant would that change be?* To formalize this, one can define a loss function $\ell(t)$ as the target function, where $\ell(\cdot)$ quantifies the model's loss in predicting a particular fact or concept. With such target functions, data attribution methods estimate the influence of each training instance on the model's alignment with the facts. The greater the counterfactual effect, the more significant the training sample is in causing the model to generate a fact, either correctly or incorrectly.

3.2.2 Landscape of Data Attribution-based Fact Tracing

A common approach in both fact tracing and concept probing methods is first to estimate the *relevance score* s_i of every training entity i in the training dataset w.r.t. a given target test data point. Then, a simple top-k selection is applied to select k data points with the highest relevance scores as the tracing/probing results. Fact tracing and concept probing have been a common downstream evaluation task for the development of data attribution methods, which we will elaborate on in Section 4.2.2.

Attribution algorithm. A common attribution algorithm for fact tracing and concept probing for LLMs is the influence function family (Akyürek et al., 2022; Wu et al., 2024b; Chang et al., 2025; Kwon et al., 2024) and its variants, such as TracIn (Lin et al., 2024b; Choenni et al., 2023, 2024b) and TRAK (Choenni et al., 2024a), where they often include additional normalization/denoising strategies (Akyürek et al., 2022; Wu et al., 2024b; Chang et al., 2025).

Influence function. Since fact tracing and concept probing are mainly used for large-scale generative models, the Grad-Dot family methods become the most feasible approach to attribute these models for their efficiency and simplicity.

In the domain of fact tracing, Akyürek et al. (2022) directly applies gradient similarity to real-world and synthetic factual datasets. While this approach achieves reasonable results, future studies (Wu et al., 2024b; Chang et al., 2025) suggest that some more normalization or denoising strategies are still needed for better approximation. Specifically, TRACK-STAR (Chang et al., 2025) proposes to normalize the raw gradients with the second moment $\mathbb{E}_x((\nabla Loss(x,\theta))^2)$ before projecting them to a lower dimension. Apart from that, it also adopts a task-specific Gauss-Newton approximation to the loss Hessian, which means that the final estimation comes from both the training and testing data distribution. While Hessian can be effective for fact tracing, Wu et al. (2024b) proposes DDA, which is a Hessian-free attribution

method aiming to debias and denoise the gradient dot product. For debiasing, it reanalyzes the effect of the false assumption of empirical risk minimization, and incorporates a correction term based on the initial state of the model. Denoising involves averaging the influence score over several epochs, which eliminates influence score discrepancies due to the overfitting or underfitting of the model during training. Finally, a contrastive influence score is computed after the debiasing and denoising procedures.

Such Gradient-Dot family methods are also widely used for concept probing. In a multilingual setting, TracIn (Choenni et al., 2024b, 2023) and TRAK (Choenni et al., 2024a) are applied to investigate how different concept dynamics, such as values shifts, data sharing, happen when fine-tuning the LLMs on different languages. A parallel line of research investigates concept dynamics in diffusion models (Xie et al., 2024a; Kwon et al., 2024), where, in particular, Xie et al. (2024a) extends TracIn to diffusion settings by incorporating gradient normalization to correct for time-step bias inherent in image generation and applying the method to image tracing.

Datamodels. MONTRAGE(Brokman et al., 2024) is a gradient-free attribution method designed specifically for diffusion models. It traces how internal representations evolve throughout diffusion models fine-tuning, and generalizes the attribution to unseen generations by training an attribution model.

Contrastive-based tracing. While TDA methods are effective at estimating counterfactual loss changes, they tend to be less effective than retrieval-based methods for tasks such as fact tracing or hallucination tracing. In order to mitigate this problem, Contrastive Error Attribution (CEA) proposes to use contrast-based tracing instead of influence-based tracing. Unlike influence functions, which estimate the counterfactual change in loss or log probability, contrastive-based tracing focuses on how the probability of the model generating a correct output is higher than that of an incorrect one. Such idea can be applied to different attribution methods, such as datamodels or gradient dot-product (Chang et al., 2025).

Iterative attribution. Han et al. (2023) applies ORCA, an iterative attribution method, to understanding the in-context learning (ICL) phenomenon in LLM pre-training. ORCA keeps track of a supportive subset within the pre-training data. During each iteration, the algorithm identifies the pre-training subset with the highest gradient cosine similarity to the ICL downstream tasks. This subset is added to the supportive set, and the model is updated using the cumulative supportive set. It turns out that the supportive subset for ICL tends to include rare tokens and is more domain-relevant, making it more challenging for LLMs.

Target function. The most common target function used in both fact tracing and concept probing is the per-sample test loss (Choenni et al., 2023, 2024a), together with normalization w.r.t. the norm of the per-sample to mitigate the outlier effects (Akyürek et al., 2022; Wu et al., 2024b; Chang et al., 2025; Kwon et al., 2024; Xie et al., 2024a).

Target subset. While the loss function is relatively fixed among studies on fact tracing and concept probing, the specific target data subsets still vary depending on different applications. One selection of the target subset is the test data on which the model produces correct prediction. This choice is commonly used in either fact tracing or concept probing, since only the acquired knowledge or concepts are trackable inside the model. For instance, Akyürek et al. (2022) suggests performing TDA on the Finetune-learned (FL) slice of the test samples. This approach ensures that the model learns the factual knowledge during the fine-tuning process rather than the pretraining. In concept probing, Choenni et al. (2024b, 2023) also focus on the set of samples that the model labeled correctly, thereby narrowing the scope of their studies to training samples that influence correct predictions.

Another approach to selecting the target subset occurs in hallucination tracing, where the goal is to identify the causes of false predictions. In this case, the target subset usually consists of the test samples on which the model makes incorrect predictions. Wu et al. (2024b) selects 4 typical hallucination types and performs contrastive data attribution based on a mixture of positive and negative test samples. Chang et al. (2025) conducts a case study on the training samples retrieved by a subset of 1,592 mislabeled test examples, attributing the incorrect predictions to model guesses based on incomplete information in the training data.

Data granularity. In the context of LLMs, example-level (Lin et al., 2024b; Akyürek et al., 2022; Wu et al., 2024b; Chang et al., 2025; Choenni et al., 2023, 2024b,a; Han et al., 2023) attribution is the most common use cases for fact tracing and concept probing, due to compute limitations. However, RapidIn (Lin et al., 2024b) enables token-level attribution by efficient projection and caching strategies.

For diffusion models, on the other hand, each individual image is the natural attribution unit (Xie et al., 2024a; Kwon et al., 2024; Brokman et al., 2024), making example-level attribution straightforward.

Scope of tracing Training a modern large-scale generative model typically involves two stages, pre-training and fine-tuning. As a result, the scope of fact tracing and concept probing can also divided into two categories: tracing the pre-training data and tracing the fine-tuning data.

Fine-tuning data. Most studies on fact-tracing and concept probing focus on attributing the fine-tuning data(Lin et al., 2024b; Akyürek et al., 2022; Wu et al., 2024b; Kwon et al., 2024; Choenni et al., 2023, 2024b,a; Han et al., 2023). Attributing fine-tuning data is more tractable due to its smaller scale, but special care must be taken, as the influence may stem not only from the fine-tuning data but also from the pre-training data. To address this, for instance, Akyürek et al. (2022) selects the finetune-learned set as the attribution targets to mitigate the influence of pre-training stage. For hallucination tracing, Wu et al. (2024b) introduces perturbations absent from the pre-training data to the fine-tuning set. Those adaptations are important for justification of the tracining procedure.

Pre-training data. Attributing the full pre-training data can be challenging, but is still tractable with suitable attributing algorithms(Xie et al., 2024a; Han et al., 2023; Chang et al., 2025). With ORCA, Han et al. (2023) attributes the in-context learning behavior of LLMs to 2.5 million pre-training instances and billions of tokens on OPT-6.7B. Chang et al. (2025) scales the influence function to use cases with over 160 billion tokens and an 8B-parameter language model by leveraging efficient gradient projection and Hessian approximation.

3.2.3 Discussion

Data attribution provides a data-centric framework for understanding how generative models acquire and express specific facts or concepts. In both fact tracing and concept probing, attribution-based methods aim to identify the training examples that causally influence a model's outputs—offering explanations grounded in the data itself. This contrasts with classical approaches that rely on textual similarity, retrieval heuristics, or probing internal representations. For example, fact tracing is often framed as a retrieval task, using methods like BM25, embedding similarity, or fine-tuned language models (e.g., T5/mT5) to locate textually relevant training data (Robertson et al., 1994; Rajani et al., 2020; Lee et al., 2024; Raffel et al., 2020; Xue et al., 2021; Menick et al., 2022). Likewise, concept probing methods such as linear classifiers or neuron activation analyses (Tenney et al., 2019; Dai et al., 2021) aim to interpret learned representations without directly attributing them to the data. While effective in certain settings, these alternatives generally measure correlation rather than causal contribution—something data attribution methods are specifically designed to capture.

By quantifying the influence of individual training examples, data attribution offers a unified lens on both fact tracing and concept probing. This is particularly effective in fine-tuning settings, where training sets are smaller and attribution becomes more tractable. However, these methods often require gradient-based computation and white-box access, limiting their scalability and applicability in black-box or large-scale pretraining contexts. A further challenge lies in the inherent mismatch between factual overlap (used in traditional fact tracing) and causal influence (captured by data attribution) (Chang et al., 2025; Wang et al., 2025a). Moreover, the ambiguous nature of "concepts" and the multi-stage training pipelines of modern models complicate attribution, making it difficult to isolate responsible training data even when full model access is available.

Unique difficulties and challenges. Despite their interpretability, most data attribution-based approaches rely on gradient-based influence estimates, which require full (white-box) model access. In real-world settings—especially for commercial or proprietary models—only black-box access is available, severely limiting applicability. Additionally, establishing ground truth for what data "caused" a model to learn a particular fact or concept is difficult in practice. This is exacerbated by the vague and often task-specific definitions of concepts and the presence of multi-stage pretraining and fine-tuning, which further obscure the traceability of learned information to specific data sources.

Empirical results further highlight the limitations of directly applying current data attribution methods to fact tracing. Extensive negative results shows that existing data attribution methods often perform worse than trivial retrieval-based baselines like BM25. These traditional methods, though rely only on lexical overlap, surprisingly outperform complex data attribution methods in identifying training examples related to factual outputs. To improve the performance of attribution methods in this area, recent work has introduced non-trivial adaptations like contrastive-based data tracing.

Nevertheless, the evaluation methodology in this domain presents several important concerns. In many existing benchmarks, the factual outputs and source inputs exhibit high lexical overlap, which inherently favors baseline methods like BM25 that rely on surface-level matching. When such overlaps are manually reduced or removed, data attribution methods often perform more competitively. For instance, Wang et al. (2025a) observed greater performance headroom for attribution methods when evaluating fact tracing under paraphrased conditions, where lexical similarity was mitigated.

Moreover, while retrieval-based evaluation is efficient and easy to implement, it may fail to reflect the true training dynamics of small language models. Park et al. (2023) demonstrated this by removing the top-retrieved data points based on either BM25, TRAK, or ground truth from the training set and retraining the model. Surprisingly, removing the TRAK-selected instances caused a larger drop in evaluation performance than removing those identified by BM25 or even those matching the ground truth. This suggests that, especially for smaller models, attribution may behave in

counterintuitive ways, and direct lexical matching between training data and factual outputs may not provide a reliable reference. Fortunately, this misalignment between human intuition and model learning behavior appears to diminish as model improves. Newer generations of LLMs tend to demonstrate more consistent attribution patterns, as shown in Chang et al. (2025), providing a better understanding of the connection between training data and model behavior.

3.2.4 Recommendation

While data attribution-based methods provide an intuitive approach to fact tracing and concept probing, their effectiveness depends on a clear dependency between the model's learned facts and concepts and its training data. If the model has already acquired the relevant knowledge during pre-training or derives it through complex reasoning, data attribution-based methods may be less effective than classical approaches (Akyürek et al., 2022). In such cases, more computationally expensive alternatives—such as training LLMs to cite corresponding sources during the pre-training phase—may be more suitable (Menick et al., 2022). Addressing these limitations and enhancing data attribution-based methods remains a crucial direction for future research.

Moreover, the current literature on data attribution is still relatively underexplored, as existing studies primarily rely on the classical loss-based attribution metrics. While it is reasonable to assume that the presence of factual knowledge or relevant concepts in training data would lead to a decrease in loss—thereby justifying the use of such metrics—the effectiveness of alternative target functions remains largely unknown. Investigating new attribution metrics and their potential impact on fact tracing is a promising avenue for future research.

3.3 Adversarial Attack and Defense

As AI systems are increasingly deployed in high-stakes applications, ensuring their robustness and safety has become a key concern. Adversarial attacks aim to exploit model vulnerabilities for malicious purposes, while defenses seek to detect and mitigate such threats. Data attribution methods—originally developed for interpretability and data-centric analysis—have found applications in both attacking and defending machine learning systems. However, due to the high computational cost of attribution algorithms, adversarial attacks have been almost exclusively explored in non-generative settings, and defense strategies for large generative models like LLMs remain relatively underdeveloped.

Despite the focus of this survey on generative models, data attribution methods have been extensively studied and applied in non-generative settings. In the contexts like image classifiers, semantic models, recommender systems, or graph neural networks, attribution techniques offer a straightforward way to assess the influence of individual data points on model behavior. Adversaries can exploit these methods to strategically insert, remove, or modify training samples, thereby manipulating model predictions. Conversely, defenders leverage attribution to detect malicious data and enhance robustness of these models. Many of these techniques can be adapted to generative models like LLMs with minimal modifications. For instance, attribution-based attacks in discriminative settings inspire analogous threats in generative AI, such as jailbreaking prompts or safety-alignment bypasses. Specifically, we also discuss in detail the attacks and defenses on a special type of learning algorithm recommendation systems, due to their similarities with LLMs. Like LLMs, modern recommendation systems increasingly rely on user-contributed data and interactive feedback loops, making them vulnerable to adversarial manipulation. By analyzing attacks and defenses in recommendation systems, we gain insights into both systematic attack methods and emerging threats in generative AI.

In the following sections, we will first explore attribution methods in non-generative models to establish foundational concepts, then extend the discussion to generative settings, highlighting key adaptations in the context of generative models.

3.3.1 Overview of Data Attribution-based Adversarial Attack and Defense

Attribution-based techniques provide a natural mechanism for both launching and defending against data-centric adversarial attacks. On the defense side, these methods can help identify harmful or toxic training examples, diagnose vulnerabilities, and sanitize training sets—often in conjunction with interpretability tools or robustness pipelines. On the attack side, the key insight is that attribution algorithms like influence functions exhibit counterfactual predictability. This allows adversaries to craft training points that systematically steer model predictions, inject biases, or interfere with model updates in a controlled way.

3.3.2 Problem Settings

Data attribution-based attacks or defenses can be applied to various realistic settings. On the attackers' side, most strategies fall under *backdoor attack*, *membership inference*, or *data poisoning*, and these methods have been used to target various machine learning domains, including recommendation systems (Wu et al., 2021; Fang et al., 2020;

Huang and Li, 2023; Wu et al., 2023a), image or semantic classification (Yang et al., 2023; Cohen and Giryes, 2024; Jagielski et al., 2021), language generation (He et al., 2024), reinforcement learning (Lobo et al., 2022), graph neural networks (Wang et al., 2024a) and machine learning services Huang et al. (2024). On the defenders' side, attribution methods majorly contribute to *data cleaning*, i.e. identifying and removing the negatively influential data points. Most efforts have focused on language modeling, particularly in areas such as toxicity detection (Han and Tsvetkov, 2020), jailbreak behavior tracing (Xie et al., 2024b; Lin et al., 2024b), and privacy protection (Liu and Yang, 2024).

On thing worth mentioning is that, while data attribution methods are powerful at launching attacks, they can also become the victims to some carefully designed adversarial attacks (Ju et al., 2022; Marchant et al., 2022; Wang et al., 2025c). These potential vulnerabilities pose threats to data economy and algorithm reliability.

Attack and defense strategies. We focus on three common attacks: (1) *backdoor attacks*, which implant triggers via input or label transformations, so that the model predicts attacker-chosen labels while behaving normally otherwise, (2) *data poisoning*, which alters the training distribution or specific samples to steer the learned model toward malicious outputs, and (3) *membership inference*, which infers whether a given example was in the training set. While these attack strategies pose threats to model safety, we also introduce data cleaning, which fix the model by detecting and removing malicious data.

Data-attribution-based backdoor attack. Backdoor attack is a kind of stealthy attack where the adversary aims to embed hidden behaviors in the model. The backdoor procedure is usually defined as two transformations on the inputs and labels, $B_{\mathcal{X}} : \mathcal{X} \to \mathcal{X}$ and $B_{\mathcal{Y}} : \mathcal{Y} \to \mathcal{Y}$. The input transformation $B_{\mathcal{X}}$ inserts a backdoor trigger, which misleads the model to predict the output as the perturbation label $y_{perturb}$. To launch an attack, the adversary will introduce the malicious inputs $((B_{\mathcal{X}}(x), B_{\mathcal{Y}}(y))$, transformed from the benign input pair (x, y), to steer the behavior of the model and implant the trigger.

To make the attack even more stealthy, camouflage samples are leveraged to hide the true intension of the adversary. Specifically, the adversary will involve another set of partially perturbed training samples $((B_{\mathcal{X}}(x), y)$ in the training set, so that the ground truth outputs remain unflipped. One application of data attribution methods on backdoor attack is to improve the design of the camouflage samples, by making them even more confusing and malicious. Using gradient ascent, camouflage samples can be updated to increase their influence on the loss on true backdoor samples. By doing so, these seemingly benign samples can be perturbed to mimic the backdoor samples, so that the true backdoor samples can be hidden.

Data-attribution-based data poisoning. Unlike backdoor attacks, data poisoning focuses on directly perturbing the model's behavior without relying on any triggering signals. Generally, we denote a generative model as $p(\cdot \mid x_0; \theta, \mathcal{X})$, where x_0 is a specific input to the model, $\mathcal{X}$ is the training distribution, and θ represents the model's final parameters. The model estimates the output distribution, which ideally follows some benign distribution $\mathcal{Y}$benign, i.e., $p(\cdot \mid x_0; \theta, \mathcal{X}) \sim \mathcal{Y}_{benign}$. The ultimate goal of the adversary is to manipulate the training distribution into $\mathcal{X}_{adv}$, such that after training, the updated model parameters θadv lead to an output distribution aligned with some malicious target $\mathcal{Y}_{malicious}$, i.e., $p(\cdot \mid x_0; \theta_{adv}, \mathcal{X}_{adv}) \sim \mathcal{Y}_{malicious}$.

In this scenario, one can use the model's loss on the malicious distribution, $\ell(y_m \mid x_0; \theta, \mathcal{X})$, where $y_m \sim \mathcal{Y}_{malicious}$, as the target function. Data attribution methods can assist in data poisoning attacks in two main ways. First, poisoning can be formulated similarly to data selection, where the adversary removes training data with benign effects and retains those that exhibit perturbative influence on the target function, estimated via counterfactual prediction. Interestingly, the counterpart of data poisoning, *data cleaning*, can also be facilitated by data attribution methods in a similar manner, which removing the perturbative data instead of the benign ones. Second, instead of removing data, the adversary may perturb selected training samples. A typical approach is to perform gradient ascent (Koh and Liang, 2017), increasing the influence of these samples on the target loss. Since such perturbations are often imperceptible to humans, poisoning through training sample modification remains stealthy and difficult to detect.

Data-attribution-based membership inference. Membership inference attacks aims to determine whether a data sample x_0 belongs to the original training set $\mathcal{X}$ of a specific model. Self-influence turns to be an effective approach to membership attacks (Cohen and Giryes, 2024), as unseen data points tend to have large influences on the loss of a test point with the same label. Consequently, if the influence of a data sample on itself is large, it is likely not used as a part of the training set of the model.

Victim models. Various machine learning models are subject to data-attribution-based adversarial attacks. The most systematic attack approaches are developed on the *recommendation systems* and *discriminative models*. With the rise of *generative models*, it turns out that many attack approaches can be transferred to this new setting, with minor adaptions.

Recommendation systems. Recommendation systems are some specific learning algorithms that are usually vulnerable to attacks due to their transparency. Notably, the sequential modeling nature of the advanced recommendation systems make them similar to generative models. In some recommendation systems, such as Amazon and Yelp, historical user ratings used to train the recommendation algorithm are publicly accessible. Besides, some recommendation algorithms are also reported, making them also exposed to the attackers. To attack a recommendation system, adversaries can inject a small number of fake users, each with carefully crafted rating histories, in order to indirectly influence the parameter updates of the system. Attacks on recommendation systems typically fall into three categories: *promotion attacks*, *demotion attacks*, and *availability attacks* (Li et al., 2016), and data attribution is widely exploited in the first two. Promotion attacks aim to promote a given item, improving its ranking relative to others in the system. On the contrary, demotion attacks pursue the opposite goal by suppressing the ranking of a target item. Both attacks can be regarded as special cases of *data poisoning*, where the injected fake data serve as toxic samples, and the intended poisoning effect corresponds to the promotion or demotion of targeted items. Availability attacks, on the other hand, aim to maximize recommendation errors and disrupt the overall recommendation algorithm. This type of attack remains relatively unexplored in the context of data attribution and falls outside the scope of this survey.

Discriminative models. Discriminative models can also fall victims to data attribution-based attacks, with the three typical classes of attacks, *backdoor attack* (Huang et al., 2024), *data poisoning* (Koh and Liang, 2017; Lobo et al., 2022; Wang et al., 2024a; Jagielski et al., 2021; He et al., 2024), and *membership inference* (Cohen and Giryes, 2024) well developed in theory.

In a backdoor attack, the adversary injects a small number of *backdoor samples* into the training dataset. These samples are crafted to induce a specific misbehavior in the model. The ultimate goal is to cause the discriminative model to misclassify any test input that contains a *backdoor trigger*, typically a imperceptible perturbation, into a target class designated by the attacker, while maintaining normal performance on clean inputs. Meanwhile, in order to conceal the attack behavior, some benign-looking *camouflage samples* may also be injected during training. Huang et al. (2024) investigate backdoor attacks in the context of Machine-Learning-as-a-Service (MLaaS), where backdoor samples are introduced through user-contributed data.

Data poisoning, on the other hand, broadly refers to the manipulation of the training data to degrade the overall performance of the model or to intentionally steer its behavior. Some data poisoning attacks are *targeted*, where the adversary aims to alter the model's behavior on a specific subset of data. For example, Jagielski et al. (2021) explore poisoning attacks that focus on a target subpopulation, while leaving the rest of the population unaffected. In contrast, *untargeted* data poisoning attacks aim to degrade the model's general performance, without targeting any particular subset, which is a more common objective in some more complicated models, such as LLMs.

Membership inference attacks (Cohen and Giryes, 2024) are another class of privacy-focused attacks that aim to determine whether a specific data sample was included in the training set of a given machine learning model. These attacks are particularly concerning in high-stakes domains, where models trained on sensitive user data. The model may leak private information through their parameters under membership inference attacks launched by adversaries.

While data attribution can be leveraged to launch attacks on machine learning models, it also plays an important role in defending them, particularly through *data cleaning*. For example, Han and Tsvetkov (2020) strengthen toxicity detectors by identifying mislabeled training samples.

Generative models. Recent studies have extended data attribution-based attacks and defenses to generative models, especially for LLMs, where safety is a growing concern. These techniques are often similar to those used for discriminative models or recommendation systems, but require key adaptations to account for the larger model scale and the sequential nature of LLMs. Most research in this area still centers on *data poisoning*, while *backdoor attacks* and *membership inference* remain relatively underexplored. He et al. (2024), for example, demonstrate that carefully selected benign data can be used to trigger toxic behaviors in LLMs.

In response to rising safety concerns, data attribution methods have also been adopted for training data sanitization in LLMs, including the detection of jailbreaking prompts and toxic examples. Xie et al. (2024b) leverage attribution techniques to identify and remove backdoor and poisoning samples from LLM training datasets.

Data attribution as victims. Interestingly, data attribution methods themselves are vulnerable to adversarial attacks, which can undermine their reliability in applications such as data valuation and machine unlearning. Wang et al. (2025c) investigate how attribution results can be manipulated even under limited adversarial knowledge. Similarly, Marchant et al. (2022) demonstrate that the computational cost of data attribution-based algorithms, such as data unlearning, can be significantly inflated by adversarial samples. Since these examples are out of the scope of the applications of data attribution, we will omit the details in the following sections.

3.3.3 Attack and Defense on Recommendation Systems

Data attribution methods are especially effective in promotion and demotion attacks for recommendation systems. The core objective of promotion and demotion attacks is to increase or decrease the ranking of a target item relative to all other items in the system, ideally affecting the preferences of as many users as possible. Taking promotion attacks as an example, the attack loss is often heuristically defined as

$$\mathcal{L}_{atk} = \sum_{i} \sum_{j \in \Gamma_{i,k}} g(\hat{r}_{i,j} - \hat{r}_{i,t}),$$

where $\hat{r}_{i,j}$ denotes the predicted rating score of user i on item j, $\Gamma_{i,k}$ represents the top-k recommendation list of user i, and g is a continuous, monotonically decreasing function, such as the Wilcoxon-Mann-Whitney loss function (Backstrom and Leskovec, 2011). By minimizing this loss, the predicted score of target item t is maximized and its ranking is improved. Demotion attacks follow a similar formulation, but with the objective reversed; thus, we omit the details here. We will introduce a few typical promotion attacks on recommendation systems in this section and summarize the details in Table 12.

Attack knowledge. To launch a successful attack on recommendation systems, attackers typically need access to the *model architecture* and *training data*. In the full-knowledge setting, the adversary knows the exact model architecture, the complete training dataset, and consequently the model parameters. While this assumption may appear strong, the attacks described above can also be extended to partial-knowledge settings, where the model is a black box and the adversary lacks full access to the training data. In practice, adversaries can often crawl partial training data from public sources on the web. Even if the exact model architecture remains unknown, the transferability of attacks enables adversaries to train a surrogate model that approximates the behavior of the target model. With white-box access to the surrogate model, the attacker can design and optimize their attack strategy, which can then transfer effectively to the real system.

Attribution algorithm. Influence functions (Koh and Liang, 2017) are the most common data attribution methods adapted for the minimization of the promotion attack loss $\mathcal{L}_{atk}$, due to their excellent counterfactual predictability. One application of influence functions is to evaluate how individual users affect the behavior of a *recommendation system*, which provides the *importance estimation* for a data record or a user. On the other hand, influence functions can also be used to estimate the impact of fake users on the *attack loss*, thereby serving as a form of *threat estimation*.

Even before the era of generative model, influence functions play a critical role in *importance estimation*. Fang et al. (2020) investigate how users' ratings influence the model's prediction on a target item t. Based on this analysis, they identify a set of *influential users* whose ratings yield the highest influence scores, and restrict the optimization of the attack loss $\mathcal{L}_{atk}$ to this subset of users, therefore improving the attack's efficiency and effectiveness. In contrast, Huang and Li (2023) focus on maximizing the estimated influence of the single-injected fake user on the model's training loss. To achieve this, they first compute influence scores for all real users and then train a three-layer neural network to predict influence scores for fake users. During optimization, the estimated influence of the fake user is jointly optimized along with the attack loss.

In the context of *threat estimation*, the application of data attribution methods are not that straightforward. Since fake users are not part of the original training set, influence functions cannot be directly applied. To address this, the influence of injecting a fake user is decomposed into two components. The first component, denoted as $\mathcal{I}_{copy}$, represents the influence of duplicating an existing real user from the system. The second component $\mathcal{I}_{pert}$, captures the additional influence introduced by perturbing the copied user into the desired fake profile. By carefully selecting the real user to copy, $\mathcal{I}_{copy}$ can be reduced to zero, allowing the estimation to focus only on $\mathcal{I}_{pert}$.

Building on this formulation, INFMIX (Wu et al., 2023a) proposes a discrete fake user profile sampling method called USERMIX, which selects fake user candidates based on their estimated influence on reducing the attack loss. TrialAttack (Wu et al., 2021) further incorporates a generative model, generative adversarial networks (GAN) into the framework to maximize the stealth of the attack. It trains a supervised model to predict the influence of fake users, aiming to maximize their effect using a GAN-based architecture. Specifically, the GAN here follows the same methodology as those used for image generation. while the discriminator attempts to distinguish fake users from real ones, the generator is trained to generate deceptive fake users that could fool the detector and also simultaneously maximize the the predicted influence of its generation. This adversarial design enhances the subtlety and effectiveness of the attack.

Notably, attacks based on heuristic attack losses remain defensible through influence function-based methods. Wu et al. (2023a) propose APT, a defense strategy that aims to "fight poison with poison." While adversarial attacks typically

inject fake users that increase the empirical risk, APT counters this by injecting specially crafted users, the ERM users, that minimize the empirical risk. The key idea is to reformulate the influence estimation by simply replacing the attack loss $\mathcal{L}_{atk}$ with the empirical risk loss $\mathcal{L}_{ER}$. This allows the defender to identify and inject users that steer the model toward adversarial robustness, effectively neutralizing the adversarial influence.

Target function. The target function of a data attribution-based attack varies depending on the specific role played by the attribution method. When used as an *importance estimator*, the target function can be defined as the predicted ranking score (Fang et al., 2020) or the overall training loss (Huang and Li, 2023). These choices aim to quantify the influence or importance of a particular rating or user with respect to the model's decision or behavior, guiding the attacker in selecting or crafting inputs that most effectively manipulate the system.

For threat estimators (Wu et al., 2023a, 2021), the target function is the *attack loss* $\mathcal{L}_{atk}$. The counterfactual of interest is the change in attack loss resulting from the insertion of a fake user. Ideally, the fake user should have a negative overall influence on the attack loss, hence increasing the likelihood that the target item is successfully promoted or demoted.

Data granularity. The granularity of data attribution on recommendation systems can differ from *per-record* to *per-user*. One single user can contribute several ratings or data records to the system, and each of them will have their own influence. Fang et al. (2020) focus on per-record influence of each single user-rating pair. It estimates the counterfactual effect of removing a single rating from a user independently, and sum up those influence as the overall influence estimate for a user. Huang and Li (2023); Wu et al. (2023a, 2021), on the other hand, directly perform per-user attribution.

Table 12: Comparison of data-attribution-based attacks and defenses on recommendation systems.

References	Attribution Method	Target Function	Granularity	Strategy
Fang et al. (2020)	IF	Predicted Ranking Score	Per-record	Data poisoning
Huang and Li (2023)	IF	Training Loss	Per-user	Data poisoning
Wu et al. (2023a)	IF	Attack Loss	Per-user	Data poisoning
Wu et al. (2021)	IF	Attack Loss	Per-user	Data poisoning
Wu et al. (2023a)	IF	Empirical Risk Loss	Per-user	Data cleaning (via injection)

3.3.4 Attack and Defense on Discriminative and Generative Models

Influence functions and their variants have proven effective in attacking a wide range of discriminative or generative models, spanning image processing, natural language processing, and even LLMs. These methods support various types of attacks, such as *backdoor attacks*, *data poisoning*, and *membership inference*. On the other hand, they are also effective for defending against attacks by identifying and removing malicious or veiled training samples, as a process known as *data cleaning*. In the following sections, we provide a detailed overview of how attribution methods are adapted and applied across these different domains, as well as a summary of representative works in Table 13.

Attack knowledge. Most of the attacks assume a white-box setting, meaning that the *model architecture*, the *training data* and hence the *model parameters* known to the adversaries (Yang et al., 2023; Cohen and Giryes, 2024; He et al., 2024). However, some studies (Jagielski et al., 2021; Huang et al., 2024) also investigate attacks launched with auxiliary data that simulates the distribution of real training data and surrogate models trained on those data.

Backdoor attacks. Huang et al. (2024) introduce UBA-inf, a stealthy yet persistent unlearning-activated backdoor attack targeting Machine-Learning-as-a-Service (MLaaS) platforms. In this setting, adversaries can upload customized data via user contribution mechanisms, and the model is periodically updated using continual learning. UBA-inf leverages influence functions to iteratively update the camouflage samples throughout the continual learning process, enhancing the concealment of the backdoor.

In each training iteration, the adversary uses influence function to estimate how perturbing a camouflage sample $\tilde{z}$ will affect the loss on the backdoor samples, by using the training loss on the backdoor samples as the target function. The camouflage samples are then perturbed in the direction that maximizes their positive influence on the backdoor objective. This influence-driven camouflage perturbation not only improves stealth of the attack, but also enables the backdoor to be more efficiently recovered during the unlearning phase, requiring fewer samples to reactivate the malicious behavior.

Data poisoning. Data poisoning is one of the most common types of attacks that can be facilitated by influence functions. Poisoning attacks can be *targeted*, aiming to manipulate the model's behavior on specific inputs. As foundational examples, Koh and Liang (2017); Yang et al. (2023) show that influence functions can be used to cause a classifier to misclassify a particular target sample by perturbing or removing training points to increase the test loss on that sample. Building on this idea, Jagielski et al. (2021) extend the attack to subpopulations, aiming to maximize the degradation of model performance on a targeted group while minimizing the impact on unrelated data. To achieve this, they introduce two filtering algorithms to identify and match samples belonging to some specific subpopulations.

In contrast, *untargeted* data poisoning aims to broadly degrade a model's overall performance rather than focusing on specific samples or subpopulations. He et al. (2024) introduce a novel approach that uses two small anchor datasets: $\mathcal{D}_{harmful}$, which contains jailbreaking samples paired with harmful instructions, and $\mathcal{D}_{safe}$, which includes the same instructions but paired with safe, aligned responses. The adversary then searches a pool of benign candidate data to find samples that exhibit high gradient similarity with $\mathcal{D}_{harmful}$ and low similarity with $\mathcal{D}_{safe}$. Surprisingly, they find that LLMs can be effectively poisoned using only benign-looking data selected through this gradient-based filtering process.

In this type of attacks, the target function is often the quantity that the attacker wants to perturb. For example, the test loss or test predictions are commonly used as target functions (Koh and Liang, 2017; Jagielski et al., 2021; Yang et al., 2023; He et al., 2024), as their gradients reveal how to degrade the model's performance on specific test instances. In the context of reinforcement learning (Lobo et al., 2022), the target function becomes the value function, guiding perturbations that directly manipulate expected returns.

Membership inference. Membership inference attacks aim to determine whether a given data sample was part of a model's training set, potentially leading to privacy breaches. In this case, self-influence serves as an effective target function. Cohen and Giryes (2024) propose a self-influence–based attack strategy, which evaluates the influence of a data point on its own. Specifically, if a sample is correctly classified and its self-influence falls within a certain range, being not too small or too large, it is considered to be a member of the training set. To account for the variability introduced by data augmentations, the self-influence is averaged over several augmented versions of the sample. This method proves highly effective, demonstrating robust performance across multiple datasets and highlighting the potential of self-influence as a powerful signal for membership inference.

Data cleaning. In contrast to adversarial uses, influence functions can also serve as an effective defense mechanism by identifying and removing poisoned or harmful training samples, a process known as *data cleaning*. This typically requires a small held-out dataset that reveals undesirable model behavior, such as misclassification (Han and Tsvetkov, 2020), jailbreaking (Xie et al., 2024b; Lin et al., 2024b), or privacy leakage (Liu and Yang, 2024). The influence of each training instance on this held-out set is computed, and samples with the highest influence scores are considered responsible for degrading the model's benign behavior.

On top of this idea, GradSafe (Xie et al., 2024b) further identifies safety-critical parameter slices in the model using cosine similarity, then computes the average gradient similarity on these parameter slices. Additionally, they propose an adaptive variant of GradSafe, which replaces the averaging with a logistic regression model trained to reweight gradient similarities across parameter slices based on their estimated importance.

Liu and Yang (2024), on the other hand, introduce Heuristically Adjusted Influence Functions (HAIF), which apply heuristic scaling to down-weight training samples with large gradient norms. This adjustment mitigates the error caused by extreme gradients and leads to more accurate identification of privacy-leaking samples.

Table 13: Comparison of data-attribution-based attacks and defenses on discriminative and generative models.

References	Attribution Method	Target Function	Granularity	Strategy
Huang et al. (2024)	`IF`	Backdoor Loss	Per-example	Backdoor Attack
Koh and Liang (2017)	`IF`	Validation Loss	Per-example	Targeted Data Poisoning
Yang et al. (2023)	`IF`	Validation Loss	Per-example	Targeted Data Poisoning
Jagielski et al. (2021)	`IF`	Validation Loss	Per-example	Targeted Data Poisoning
He et al. (2024)	`Grad-Similarity`	Validation Loss	Per-example	Untargeted Data Poisoning
Cohen and Giryes (2024)	`IF`	Self-Influence	Per-example	Membership Inference
Han and Tsvetkov (2020)	`IF`	Validation Loss	Per-example	Data Cleaning
Xie et al. (2024b)	`Grad-Similarity`	Validation Loss	Per-example	Data Cleaning
Lin et al. (2024b)	`IF`	Validation Loss	Per-example	Data Cleaning
Liu and Yang (2024)	`HAIF`	Validation Loss	Per-example	Data Cleaning

3.3.5 Discussion

Despite the growing urgency to secure generative AI systems, especially LLMs, data attribution methods remain underutilized in adversarial settings. Most attribution-based strategies have been developed for data poisoning attacks and dataset sanitization in non-generative, discriminative models, largely due to their high computational cost and reliance on white-box access. These requirements make them ill-suited for large-scale generative models, where training data is vast, access is restricted, and real-time responsiveness is essential.

In contrast, classical adversarial approaches have rapidly matured across both attack and defense. Black-box attacks often exploit prompt engineering, such as paraphrasing or template perturbation, to trigger undesired behaviors without model access (Jin et al., 2020; Yi et al., 2024). White-box strategies further optimize prompts or train surrogate models to design highly transferable adversarial inputs (Zou et al., 2023; Zhang et al., 2023c; Qi et al., 2024). On the defense side, black-box methods include prompt filtering and randomized perturbations (Jain et al., 2023; Cao et al., 2024a), while white-box defenses use fine-tuning, interpretability tools, or auxiliary models to detect unsafe inputs (Bianchi et al., 2024; Xie et al., 2024b; Zhang et al., 2025d). These methods have proven to be more efficient and scalable than attribution-based ones, particularly in real-world deployments.

That said, data attribution offers unique benefits not captured by these classical techniques. By tracing model behaviors back to specific training examples, attribution enables causally grounded analyses of adversarial vulnerabilities and a principled way to clean or curate datasets. This is especially valuable for defense: identifying toxic, biased, or jailbreaking-prone data can lead to interpretable and targeted mitigation strategies. While attribution-based attacks remain largely limited to toy or fine-tuning-scale models, their interpretability and data-centric perspective provide a complementary angle to existing adversarial toolkits.

Finally, we remark that this section focuses on the *application* of attribution methods for adversarial attack and defense. A distinct but growing body of work aims to *attack attribution methods themselves*, e.g., by misleading influence estimates (Ju et al., 2022; Wang et al., 2025c), which we leave outside the scope of this survey.

Unique difficulties and challenges. Despite their interpretability and data-centric appeal, attribution-based adversarial methods face several fundamental limitations in generative settings. Most rely on computationally intensive influence estimation techniques involving higher-order gradients, which scale poorly to modern LLMs. These methods also assume white-box access to model parameters and training procedures—an unrealistic requirement for proprietary or production-scale systems. Additionally, the long-range dependencies and multi-stage training pipelines characteristic of generative models make it difficult to attribute specific behaviors to individual training examples. Even when technically feasible, the resulting signals may be too diffuse or unstable to support reliable attack or defense. From a practical standpoint, attribution-based methods currently lack the scalability, flexibility, and robustness of classical adversarial techniques and are unlikely to match them in providing certified guarantees against adaptive attacks.

3.3.6 Recommendation

Looking forward, the utility of data attribution in adversarial settings may lie more in defense than in attack. While attribution-based attacks face significant scalability and access constraints, emerging work on black-box influence estimation (Jiao et al., 2024) suggests it may be possible to design lightweight defenses that flag or filter harmful training data without full model access. Integrating such methods into generative training pipelines could improve transparency and safety with minimal overhead. Nonetheless, attribution is unlikely to replace conventional adversarial techniques; its primary contribution may instead be as a diagnostic or forensic tool, offering interpretable insights into the data-driven roots of model vulnerabilities.

3.4 Data Economy

Data economy is "a global digital ecosystem where data is gathered, organized, and exchanged to create economic value" (Sestino et al., 2025). It demands the development of new methods for valuing data, ensuring fair compensation for its use, and designing equitable data markets. The rise of generative AI has intensified challenges related to data economy, affecting industries such as music, media, and beyond.

3.4.1 Overview of Data Attribution-based Data Economy

Data attribution in the context of data economy aims to address two key questions:

1. How can we fairly assign credit to data contributors whose data was used during training?
2. How can we construct equitable, collaborative, and reliable data markets based on these contributions?

To address (1), data attribution methods such as Data Shapley (Ghorbani and Zou, 2019) and influence functions (Koh and Liang, 2017) are employed. These methods assign a value to each individual data point based on its contribution to a predefined utility function-a process known as *data valuation*. Building on this, (2) involves designing allocation rules based on the valuation results, enabling data contributors to be rewarded through revenue sharing, data incentives, and more, thereby fostering a sustainable and mutually beneficial data economy. We will call this process of stimulation and compensation as *data compensation*.

3.4.2 Problem Settings

As discussed in the previous section, *data valuation* and *data compensation* are two core research direction in the context of data economy.

Data valuation. *Data valuation* assigns a single value to quantify the contribution of an individual data point to a utility function v, where v can be any metric of interest, such as empirical risk or accuracy. An ideal data valuation method should be *equitable*, satisfying the four desirable properties defined in Section 2.3.1: **Null Player**, **Symmetry**, **Additivity**, and **Efficiency**.

Data attribution scores provide a natural framework for quantifying data valuation. Among all attribution methods, data Shapley (Ghorbani and Zou, 2019) and its variants are the most common algorithm used for data valuation. They adopt a cooperative game theory perspective and naturally satisfy all four properties. Gradient-based methods, such as influence functions, are also widely used due to their direct connection to the utility function and their computational efficiency. However, they are not fully *equitable*. Due to their leave-one-out nature, most influence function methods fail to satisfy **Additivity** and **Efficiency**.

Data compensation. While *data valuation* identifies the importance of each data sample, *data compensation* focuses on translating these valuations into fair and practical rewards for data contributors. Compensation is a necessary step, as data value does not directly correspond to actual rewards, due to complex market dynamics and the buyer-seller relationship in data economy. A typical example is the free data market. Zhang et al. (2025b) systematically shows that without appropriate pricing mechanisms, the outcome can end up with a lose-lose situation for both buyers and sellers.

The complexity of the market is further reflected in the diversity of revenue sources and the various forms that rewards can take. In the domain of generative AI, data valuation methods can attribute specific generations back to copyrighted training samples (Deng et al., 2024b; Wang et al., 2024c). However, determining fair compensation remains ambiguous, as revenue may originate from various sources, and generated content may differ in popularity. Moreover, rewards can take different forms. For instance, Tay et al. (2022) explores data rewards, where contributors receive synthesized data generated by a model collaboratively trained on all contributors' data.

Privacy and fairness are also critical concerns in the compensation process. Data valuation typically requires access to the underlying data, which introduces risks such as data leakage or theft by buyers. Conversely, revealing the valuation algorithm to sellers may expose the system to manipulation. To address these issues, Tian et al. (2022) proposes a framework that enables secure data valuation while ensuring proper compensation for data providers.

3.4.3 Data Valuation and Compensation in the Data Economy

Attribution algorithms. Data Shapley (Tian et al., 2022) and variants of influence function, such as TracIn (Pruthi et al., 2020), TRAK (Park et al., 2023) are the most common algorithms used in data economy. In addition to assigning a value or score to each data point, they can also be leveraged to ensure the fair distribution of the rewards.

Data valuation. The most straightforward approach to data valuation is through Data Shapley ϕ_i (Ghorbani and Zou, 2019; Wang et al., 2024c; Tian et al., 2022; Tay et al., 2022), which is originally approximated via Monte Carlo sampling. From a game-theoretic perspective, Data Shapley distributes the total reward for the collective contribution of all data points in an equitable manner. Recent works, such as in-run Shapley value estimation (Wang et al., 2025a), have further improved the scalability of this method for large models. In addition, Tian et al. (2022) proposes training a set function model on a small subset of pre-computed Shapley values, which can then be used to predict the value of the remaining data points efficiently.

Other methods, such as influence functions, while not strictly *equitable*, are still widely used in data valuation due to their strong counterfactual estimation capabilities. For example, Deng et al. (2024b) employ TracIn and TRAK to trace generative AI outputs in music back to the original copyrighted training samples, demonstrating both the consistency and practical utility of these techniques. In addition, model-agnostic approaches have also been explored for data valuation. LAVA (Just et al., 2023), for instance, assigns value to each data sample based on its sensitivity to the class-wise Wasserstein distance between training and validation sets. This approach indirectly links each training sample to validation performance, offering a flexible and scalable alternative without dependency on any specific model.

Data compensation. In some of the cases, data compensation can be simply done in proportion to the data values (Wang et al., 2024c), which basically ensures the fairness. In free market settings, where things are complicated by the buyer-seller relationship, Zhang et al. (2025b) highlights the risks of exploitative pricing and advocates for setting the price based on the buyer's maximum willingness to pay (MWP), which is a value determined by both the marginal utility gain from the acquired data and the buyer's budget surplus.

Another interesting case of data compensation is distributing data rewards. In the setting of Collaborative Generative Modeling (CGM), the goal is to train a generative model using datasets $\mathcal{D}_{1,\dots,N}$ provided by N contributors and to fairly allocate synthetic data rewards sampled from the model back to those contributors. Data Shapley can be integrated with Maximum Mean Discrepancy (MMD) (Gretton et al., 2012), a statistical metric that quantifies the difference between two distributions, to guide fair compensation. For each contributor i, a larger MMD-based data value $v(\mathcal{D}_i)$ is assigned if the distribution of their data is more aligned with the joint distribution used for training and the synthetic samples, i.e., if the MMD between their data distribution and the global distribution is small. This indicates that their data plays a more central role in the generation of synthetic outputs. The final upper bound on reward for contributor i is then computed as

$$r_i = v(\mathcal{D}_i) \cdot \left(\frac{\phi_i}{\max_j(\phi_j)} \right)^{\rho},$$

where ϕ_i is the Shapley value of contributor i , normalized by the maximum Shapley value across all contributors, and ρ is a tunable parameter controlling reward sensitivity to Shapley values. During the reward allocation phase, synthetic data is randomly sampled and added to the reward pool $\mathcal{R}_i$ for contributor i, until their updated data value $v(\mathcal{D}_i \cup \mathcal{R}_i)$ exceeds the bound r_i. This approach ensures that each contributor receives synthetic data that is beneficial to them and that larger contributions are rewarded more generously—promoting both fairness and utility in the data economy.

Privacy and security are additional concerns in the compensation process. Tian et al. (2022) propose a framework where the data remains encrypted to the buyer until payment is completed, which enhances data security and ensures fairness in compensation. Since the specifics of the encryption scheme fall outside the scope of this survey, we omit the technical details here.

Target function. The target function used in Data Shapley algorithms are often referred to as the utility function, which can take various forms depending on the application. In the context of generative AI copyright tracing, a common choice is the *log-likelihood* of a given generation (Deng et al., 2024b; Wang et al., 2024c), which evaluates the extent to which specific copyrighted training samples contribute to the generated output. Alternatively, distributional metrics can also serve as the utility function. For instance, Tay et al. (2022) adopt an MMD-based score to quantify each contributor's influence on the overall distribution learned by the generative model. In other settings, standard performance metrics (e.g., accuracy or loss on a validation set) may also be used as the utility target.

Data granularity. The granularity of attribution in data valuation or compensation can be either per-example or per-dataset. In practice, data contributors typically provide an entire dataset to the buyer, while Data Shapley is computed at the level of individual data points, as they represent the fundamental units of contribution. Due to the **Additivity** property of Shapley values, this two settings have almost no difference, as the value of a dataset is simply the sum of the Shapley values of its constituent data points.

3.4.4 Discussion

Classical approaches in data economy and AI often assess data through a combination of intrinsic and extrinsic factors. Raskar et al. (2019) note that "data valuation use-cases can be performed via intrinsic factors of evaluation such as based on quality of information within the dataset or via extrinsic factors of evaluation such as based on demand-supply, market economics, game-theoretic mechanisms and speculative market forces, or via a combination of both." The first part of this statement aligns with techniques in *data valuation*, which aim to reveal the inherent value or utility of the data. The latter part reflects the practical considerations involved in *data compensation*, where contributors are rewarded not only according to intrinsic value but also based on market dynamics such as demand and supply. We refer interested readers to recent surveys for a broader overview of this topic (Liang et al., 2018; Sim et al., 2022; Zhang et al., 2023b).

Data attribution methods stand out among data assessment approaches due to their ability to provide valuable insights into the training mechanisms. These methods are often grounded in solid theoretical foundations, offering principled and quantifiable measures of data contributions through frameworks such as Shapley values. As a result, compensation based on attribution methods is generally considered more equitable and fair. In contrast, the emerging paradigm of "data labor" reconceptualizes user data from free capital to recognized labor. In this model, users would be rewarded for

their contributions, potentially improving data quality, encouraging innovation, and providing a more equitable share of value (Arrieta-Ibarra et al., 2018). However, these traditional approaches often rely on heuristic assessments, such as the difficulty of data acquisition or market-driven pricing. These methods lack principled guarantees, and when applied exploitatively, they risk undermining the long-term sustainability of the data economy (Zhang et al., 2025b).

Unique difficulty and challenges. Data attribution methods, while theoretically robust, can be prone to adversarial attacks (Wang et al., 2025c). This introduces vulnerabilities into data valuation and marketplace systems. Moreover, data attribution methods typically require direct access to training data—a condition that is often unmet in large generative AI systems.

3.4.5 Recommendation

When training data is accessible, data attribution methods are particularly suited for assessing and compensating data contributions, as they offer theoretically grounded and equitable allocations, through frameworks such as Shapley values and influence functions. These methods enable a principled understanding of each data point's impact on the model's performance, making them a natural choice for fair reward distribution in collaborative learning settings.

However, in scenarios where training data is not directly accessible—either due to privacy constraints, security concerns, or technical limitations—classical approaches may be more practical. These methods typically rely on extrinsic signals such as market demand, acquisition cost, or supply scarcity, and while they lack formal fairness guarantees, they can still provide useful heuristics for data valuation and pricing in real-world markets.

3.5 Other Applications

While core applications of data attribution—such as data selection, fact tracing and concept probing, adversarial attack and defense, as well as data economy—have received growing attention, several emerging domains have begun to explore their potential. These newer applications, though less mature, highlight the versatility of data attribution and suggest promising future directions. In this section, we discuss three such areas: artifact detection and bias mitigation, memorization and privacy leakage detection, and machine unlearning. For each, we define the problem setting and examine how attribution techniques help address its central challenges.

3.5.1 Artifact Detection and Bias Mitigation

Problem setting. Modern ML systems are vulnerable to learning spurious patterns or encoding social and structural biases embedded in their training data. These artifacts, such as confounding visual cues in medical imaging or demographic correlations in NLP, can degrade generalization and produce unfair behavior. A central challenge is to identify the specific training examples that introduce such undesirable behaviors and to develop mechanisms to remove, down-weight, or mitigate their influence. This requires both fine-grained accountability and task-specific interpretability.

Data-attribution-based artifact detection and bias mitigation. Data attribution provides a principled framework for tracing model behavior back to individual training examples. The effectiveness of this approach hinges on two components: (1) the design of an appropriate *target function* to measure the undesirable behavior, and (2) a mitigation strategy that acts on the attribution scores computed with respect to that function.

Artifact detection. In artifact detection, the goal is to isolate confounding examples that undermine robustness: typically, those that lead to *incorrect predictions* or abnormally high confidence on carefully chosen validation instances. By leveraging attribution methods' ability to estimate counterfactuals, researchers can assess how the inclusion or removal of a specific training example affects model performance on a validation set. A common target function for this purpose is the *validation loss* on test instances suspected of being influenced by artifacts. Attribution scores with respect to this target reveal whether a training point contributes to misbehavior, such as reduced prediction confidence or spurious performance gains.

We highlight two representative works that adopt this paradigm. An early example is Han et al. (2020), which focuses on NLP classification tasks such as sentiment analysis and natural language inference. They apply influence functions using validation loss as the target function and examine the training points with the highest attribution scores. As an early exploration, their approach is straightforward: they analyze which training examples most increase validation loss when removed, identifying potential artifacts via simple ranking.

A more recent study by Pezeshkpour et al. (2022) considers large-scale text generation tasks and proposes a more fine-grained attribution method. They first compute instance-level influence scores using validation loss on a hand-

selected test set as the target. Then, to localize artifacts within each example, they combine data attribution with *feature attribution*: they compute input gradient-based saliency maps by taking the gradient of the influence score with respect to each input token. This two-step approach allows them to not only identify problematic training samples, but also pinpoint the specific *token-level* features responsible for the artifact.

Bias mitigation. In bias mitigation, the goal is to reduce performance disparities across demographic or structural groups in the dataset. This typically involves minimizing discrepancies in group-based metrics such as true positive rate (TPR), false negative rate (FNR), or other fairness-specific objectives. Data attribution offers a principled way to simulate counterfactuals: given a well-defined target function that captures fairness notions, attribution methods can estimate which training examples most contribute to group disparities. A key distinction from artifact detection is that bias mitigation not only seeks to *identify* biased or harmful examples but also to *modify* the training process in response through reweighting, relabeling, or removing those examples.

Although this direction remains underexplored for generative models in particular, several recent works demonstrate how data attribution can support fairness improvements in discriminative settings.

Two recent methods, D3M (Jain et al., 2024) and FairIF (Wang et al., 2024b), use data attribution to improve group-level performance without requiring manual intervention. D3M applies `TRAK`-based attribution to identify subsets of training data that disproportionately impact under-performing groups. The target function is group-aggregated test loss, and group-level attribution is used to determine which examples most exacerbate fairness gaps; this information is then used to rebalance the training data. In contrast, FairIF employs a two-stage pipeline. In the first stage, it uses influence functions with group-specific target functions, such as TPR and TNR within each subgroup, to estimate the influence of training examples on fairness objectives. In the second stage, the estimated influence scores are used to compute reweighting coefficients, and the model is retrained using this reweighted dataset. This approach improves fairness in a data-driven and systematic manner.

In the NLP domain, Brunet et al. (2019) studies social bias in static word embeddings such as Word2Vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014). Using influence functions, they attribute changes in the WEAT (Word Embedding Association Test) score (Caliskan et al., 2017)—a common metric for measuring bias in embeddings—to individual training documents. By ranking documents by their influence on WEAT, the method identifies high-impact sources of bias and enables targeted removal or curation. This represents a natural application of attribution to unsupervised or non-classification models.

A related direction concerns biases that arise not from group-based disparities but from train-test distribution mismatch. This scenario resembles data cleaning (see Section 3.1) but involves a more active *mitigation* step. Kong et al. (2022) explores this case by proposing a novel use of influence functions to actively *correct* training labels rather than discard samples. Here, the target function is the standard validation loss, and attribution scores estimate the effect of relabeling each training point. Based on this, the authors design a relabeling function that selects the optimal label under counterfactual analysis, thereby mitigating bias while preserving valuable examples and improving model robustness.

Discussion. Data attribution offers a flexible and principled toolkit for detecting artifacts and mitigating bias by identifying and intervening on influential training examples. Its core strength lies in simulating counterfactual effects: attribution methods can estimate how small changes in the training data impact downstream behavior, enabling targeted interventions. Crucially, this counterfactual perspective is adaptable to a wide range of settings and objectives—from robustness to fairness—simply by choosing an appropriate target function. This makes data attribution a unifying framework for a broad class of problems involving harmful data influence.

Nonetheless, current applications of attribution-based artifact detection and bias mitigation are largely confined to supervised, discriminative models and are typically applied in small- to medium-scale settings. Extending these approaches to large-scale or generative models remains a significant open challenge. From a data attribution perspective, future progress in this area may require:

- **Target function design**: Developing principled target functions that reflect nuanced forms of bias or artifact-driven behavior, particularly for open-ended outputs. For instance, instead of standard loss, one might consider diversity, toxicity, or factuality metrics in language generation.
- **Granular attribution**: Leveraging finer-grained attribution units (e.g., token spans, sentence pairs, visual regions) to localize problematic features within training examples, as seen in the hybrid data-feature attribution approach by Pezeshkpour et al. (2022).
- **Intervention strategies**: Expanding beyond removal and reweighting to include soft interventions such as relabeling, synthetic data augmentation, or guided training-time regularization informed by attribution scores.

As foundational models grow in scale and scope, so too does the risk of learning from biased or confounded data. Data attribution stands out as a promising direction for making these systems more interpretable, robust, and fair. However, significant methodological advances are still needed to realize its full potential in generative and large-scale settings.

3.5.2 Memorization Detection

Problem setting. Large generative models are prone to memorizing parts of their training data (Morris et al., 2025a), largely due to their high capacity and over-parameterization. While some level of benign memorization is expected, such as well-known phrases or public content, uncontrolled memorization poses risks to generalization, privacy, and robustness. The goal of memorization detection is to identify whether a model has memorized specific training examples, and if so, to determine which examples contributed most to this behavior.

Data-attribution-based memorization detection. Data attribution offers a causal lens for understanding memorization by linking generated outputs back to influential training examples. A central challenge is defining what constitutes "memorization" in a way that is measurable and aligned with model behavior. Early work by Feldman and Zhang (2020) introduces a formal definition of *label memorization* in supervised classification. Specifically, the memorization of a training example $z_i = (x_i, y_i) \in D_{\text{raw}}$ under training algorithm $\mathcal{A}$ is defined as

$$\text{mem}(\mathcal{A}, D_{\text{raw}}, z_i) \coloneqq \Pr_{h \leftarrow \mathcal{A}(D_{\text{raw}})}(h(x_i) = y_i) - \Pr_{h \leftarrow \mathcal{A}(D_{\text{raw}} \setminus \{z_i\})}(h(x_i) = y_i).$$

This counterfactual definition aligns naturally with data attribution: by setting the target function to the model's prediction confidence on z_i, self-influence (see Section 3.1) becomes a direct estimator of memorization.

In generative settings, however, applications of attribution remain limited and often heuristic. For example, Liu and Yang (2024) applies influence functions to detect privacy leakage by treating the validation loss on a held-out test sample as the target function. Similarly, Wang et al. (2025d) employs TracIn in a comparable setup. While these approaches are practical, the use of validation loss as a proxy for memorization lacks theoretical justification, making these applications relatively shallow. A more principled connection between generative memorization and attribution remains to be established.

Discussion. Unlike similarity-based memorization detection methods, attribution techniques offer more precise diagnostics by not only identifying whether memorization has occurred but also revealing which training examples are causally responsible. This is particularly valuable for debugging training data, auditing privacy risks, or performing targeted unlearning.

However, a key limitation lies in the absence of a formal and community-accepted definition of *harmful* memorization, especially in the context of generative AI. Without such a definition, attribution-based methods risk being applied in ad hoc or heuristic ways. To advance this direction, future work should aim to formalize memorization metrics for generative models—potentially drawing from the existing literature (Morris et al., 2025b)—and integrate attribution tools into training-time pipelines for early detection and proactive mitigation.

3.5.3 Machine Unlearning

Problem setting. Machine unlearning aims to remove the influence or effects of specific training examples from a model. This capability is essential for complying with data deletion requests (e.g., under privacy regulations (Mantelero, 2013)) to protect sensitive data, or to erase unwanted knowledge that is accidentally acquired by the model during training. While retraining from scratch is the most principled form of "exact unlearning," it is typically infeasible in practice due to high computational costs. As a result, there is growing interest in efficient *approximate unlearning* methods that can remove the impact of selected training points without full retraining.

A key challenge lies in the *global* nature of unlearning: the objective is not merely to alter predictions on a few test points, but to modify the model parameters such that the overall behavior reflects the absence of the removed data. With the growing complexity of modern model training setups, accurately estimating the counterfactual effect on model parameters under general settings (e.g., non-convexity, inherent randomness, etc.) remains a significant technical obstacle.

Data-attribution-based machine unlearning. Since unlearning is fundamentally about removing the influence of specific training examples, it naturally connects with data attribution methods. In convex settings, this connection has been explicitly explored: early certified unlearning algorithms for convex models effectively employ influence-function-style updates. For instance, Guo et al. (2020) derives a closed-form parameter update via *one Newton step* for removing

one sample z, given by

$$\hat{\theta}_{-z} \approx \hat{\theta} + \frac{1}{n} H_{\hat{\theta}}^{-1} \nabla \ell(z; \hat{\theta}),$$

which mirrors the classical influence function approximation (Equation (2)) for the parameter change induced by removing a training example z. Similarly, Neel et al. (2021) develops descent-based methods grounded in similar first-order influence approximations. However, these approaches are limited by their reliance on convexity and smoothness, which restricts their applicability to modern deep models, where such approximations can be inaccurate or unstable.

Recent work has begun to address this gap by leveraging advances in data attribution for complex, non-convex models. In particular, Georgiev et al. (2025) proposes an unlearning method that uses counterfactual influence estimates, as computed by TRAK (Park et al., 2023), to guide targeted model updates. Instead of retraining the model from scratch, they identify and modify only the parameters most affected by the example to be unlearned, enabling more efficient and localized unlearning. While their method does not provide formal guarantees, they employ their proposed algorithm on ResNet models on image classification tasks, demonstrating that attribution-informed unlearning is empirically viable even in the absence of convexity.[15]

Discussion. Attribution-based unlearning has shown its potential in the classical classification settings, but in general, this direction remains largely unexplored, let alone in the context of large generative models. Several obstacles hinder progress: high computational costs, the challenge of isolating a training point's impact on open-ended outputs, and the lack of robust approximations for global parameter influence with guarantees. For instance, the influence function suffers from non-convex loss functions and multiple local minima of model checkpoints, hence can not accurately predict the effects of removing a data point for a *single model*, but only an expectation over training randomness.

A promising future direction is to focus on designing scalable surrogates for attribution that are tailored to the needs of generative models, and also leverage recent dynamic attribution methods that are more *single-model*-focused methods to overcome the training progress.

4 Evaluation

In this section, we present common approaches for evaluating data attribution methods. We categorize these evaluations into four groups: (i) counterfactual prediction of model behavior, (ii) utility/effectiveness in downstream tasks, (iii) computational costs, and (iv) other miscellaneous criteria. For each group, we introduce the relevant definitions and highlight general limitations. We also discuss specific considerations for generative models, including large language models and diffusion models. Finally, we provide suggestions and recommendations for developing improved evaluation methods for future work.

4.1 Counterfactual Prediction of Model Behavior

Notations. Consider a training dataset $D = \{z_1, z_2, \ldots, z_n\}$, where each z_i is a training example. For any data subset $S \subseteq D$, let $\hat{\theta}_S$ denote the (optimal) model obtained by minimizing the following empirical risk over S:

$$\hat{\theta}_S = \arg\min_{\theta} \mathcal{L}(S; \theta) = \arg\min_{\theta} \frac{1}{|S|} \sum_{z_i \in S} \ell(z_i; \theta),$$

where $\ell(z_i; \theta)$ is the loss incurred by the model with parameters θ on training data z_i. We further define $f(z; \theta)$ as the target function, which quantitatively represents the *model behavior* when the corresponding model is evaluated at a certain sample z in interest. Note that the input to f can also be a data subset (e.g., $f(S; \theta)$). In particular, for a query/test dataset $D_{\text{eval}} = \{z_{\text{test}}^{(j)}\}$ in interest, we may write $f(D_{\text{eval}}; \theta)$. We also denote an attribution method as τ. For a given attribution method τ applied on model θ, a training dataset S and the target function $f(D_{\text{eval}}; \theta)$ associated with a test dataset D_{eval}, we define the attribution score associated with training example z_i as $\tau(z_i, S; f(D_{\text{eval}}; \theta))$.

Overview. For a given data attribution method τ, the (optimal) model $\hat{\theta}_D$ trained on full training dataset D and a fixed query dataset D_{eval}, evaluation methods in this group examine how well (i) the *induced attribution scores of training samples*: $\left\{\tau(z_i, D; f(D_{\text{eval}}; \hat{\theta}_D)), \forall z_i \in D\right\}$ and (ii) the *counterfactual predictions of model behavior* when the model is directly re-trained on counterfactual subsets $S_{z_i} : \left\{f(D_{\text{eval}}; \hat{\theta}_{S_{z_i}}), \forall z_i \in D\right\}$ align with each other.[16] Typically,

[15] It is worth noting that in Georgiev et al. (2025), they have also provided convergence theory in the case of linear models, echoing Guo et al. (2020); Neel et al. (2021).

[16] Note that the definition of counterfactual dataset S_{z_i} depends on each training example z_i.

these evaluation methods will (i) apply data attribution methods to compute the attribution scores for each training data point z_i, (ii) compute certain types of counterfactual predictions as **ground-truth** values (such as direct model output $f(D_{\text{eval}}; \hat{\theta}_S)$ or output of different target functions) and (iii) compute the *goodness score* between the attribution scores and the **ground-truth** values. For example, the *goodness score* can be defined as some correlation coefficients, measuring the correlation between two lists of attribution scores and *ground-truths*.

Below, we introduce some important definition of the **ground-truth** values, where their corresponding *goodness score* will be discussed in each subsection. For each group of evaluation methods, we also provide discussion on its general limitation and applicability to large generative models.

4.1.1 Leave-One-Out (LOO) Influence

Leave-One-Out (LOO) influence (Cook and Weisberg, 1982; Koh and Liang, 2017) evaluates how much a model's performance changes when a single data point is removed from the training set. Specifically, models are trained with each data point dropped one at a time, and the performance (i.e. the value of target function output) is measured to observe the effect of excluding that particular point. Formally, given a test example z_{test} in interest, LOO influence defines the *ground-truth* for training sample z_i as $f(z_{\text{test}}, \hat{\theta}_{D-\{z_i\}})$, where the counterfactual data subset $S \triangleq D - \{z_i\}$ is the full dataset with z_i removed. For the target function, a common choice is the loss function $\ell(z_{\text{test}}; \theta)$. Thus, the corresponding LOO influence can be written as $\mathcal{I}_{\text{LOO}}(z_i) =: \ell(z_{\text{test}}; \hat{\theta}_{D-\{z_i\}}) - \ell(z_{\text{test}}; \hat{\theta}_D)$. For the corresponding *goodness score*, the Pearson correlation coefficient (Pearson, 1895) is used by comparing the attribution scores with the change in model behavior observed when that data point is removed. Representative experiment setups with LLO influence are summarized in Table 14.

Influence-function-based influence. As an approximation of LOO influence, influence function (Koh and Liang, 2017) exhibits strong correlation with LOO ground-truths under certain conditions, such as the convergence of model and convexity of the loss function. However, for papers that propose variants of influence functions (Koh and Liang, 2017), LOO influence can not capture the most accurate information since those conditions may not be satisfied. Thus, these papers evaluate the quality of their proposed data attribution methods by treating the attribution scores derived from either exact or approximate of influence function as ground-truths. For example, FastIF (Guo et al., 2021) and DataInf (Kwon et al., 2024) compare their methods to the exact computation or LiSSA-based approximation of influence function and conduct correlation analysis (via Pearson correlation coefficient (Pearson, 1895) computation).

Trajectory-specific LOO influence. To overcome the limitation of typical LOO influence, which assumes permutation invariance of training data and only consider the final state of the trained model, trajectory-specific LOO influence (TSLOO influence) (Hara et al., 2019; Wang et al., 2025b) has been developed to quantify the change in loss when a specific data point is removed during an iteration within a specific training run. Specifically, a training run is defined as a sequence of mini-batches and the random initialization that is defined and fixed before model training. The advantage of TSLOO influence is that it considers the case that an identical data can have significantly different attribution scores when introduced in different training stages. Correlation analysis has been carried out with epoch-specific single data removal influences (i.e. each data point within a specific epoch or all epochs is removed) as ground-truths (Hara et al., 2019; Wang et al., 2025b).

Table 14: Representative experiment settings with LOO influence.

Reference	Models	Datasets
Language Models		
Wang et al. (2025b)	Pythia-410M (Biderman et al., 2023)	Pile (Gao et al., 2020)
Jiao et al. (2024)	Pythia-1B (Biderman et al., 2023)	Alpaca (Taori et al., 2023), K-Means-100 (Li et al., 2024c)
Diffusion Models		
Dai and Gifford (2023)	DDPM (Ho et al., 2020b)	MNIST (LeCun et al., 1998), CIFAR-10 (Krizhevsky, 2009), CelebA (Liu et al., 2015)

Limitations. While LOO influence provides a theoretically grounded approach to measuring the effect of individual training examples, it comes with several practical and theoretical limitations:

- **Assumption of unique solution:** LOO influence typically assumes that the solution to the underlying loss minimization problem is unique, requiring a fixed reference point to compute the effect of data removal. However, large and deep neural networks often violate this assumption, making LOO estimates ambiguous (Basu et al., 2021; Bae et al., 2022; Schioppa et al., 2023; Bae et al., 2024).
- **Indistinguishability:** In modern large-scale datasets, models trained with only one data point removed are often indistinguishable with the original model. This ambiguity persists even when averaging over multiple independent re-trained models with different random initializations (Nguyen et al., 2023).
- **Fragility:** Several studies (K and Søgaard, 2021; Epifano et al., 2023; Basu et al., 2021; Bae et al., 2022, 2024) have highlighted the fragility of LOO influence. Notably, Bae et al. (2022) identifies failure modes of influence-function-based estimates and introduces the proximal Bregman response function (PBRF) as a more reliable ground-truth.
- **Computational inefficiency:** LOO influence requires retraining the model n times to compute influence scores for all training samples, where n is the number of training instances. This makes it computationally infeasible for large datasets.

Generative model settings. Computing exact LOO influence is already impossible for large models before the era of large generative models. As an efficient alternative, influence-function-based influence approximates LOO influence by certain technical assumption. Nevertheless, it is still computationally infeasible for large generative models mainly due to the expensive computation of Hessian matrix and inverse Hessian-vector-product. For example, most papers (Koh and Liang, 2017; Ko et al., 2024; Kwon et al., 2024) evaluating their method with influence-function-based influence typically only consider small models, such as MLP/CNN or RoBERTa-large (Liu et al., 2019), whose sizes are much smaller than modern generative models.

4.1.2 Linear Datamodeling Score (LDS)

Proposed by Park et al. (2023), the linear datamodeling score (LDS) evaluates data attribution methods by probing their ability to estimate the counterfactual predictions of models being re-trained on random data subsets. By the additivity assumption on attribution methods [17], LDS defines a method's counterfactual prediction as the sum of the scores of the examples in the given training subset. Formally, for a given attribution method τ and a test example z_{test} in interest, we can first define the *attribution-based prediction* of the target function $f(z_{\text{test}}; \theta)$ as follows:

$$g_\tau(z_{\text{test}}, S; D) \triangleq \sum_{i: z_i \in S} \tau(z_i, S; f(z_{\text{test}}; \hat{\theta}_S)),$$

where D is the full training set and $S \subseteq D$ is a subset of D. Conceptually, $g_\tau(z_{\text{test}}, S; D)$ computes the overall attribution of the subset S on the test example z_{test}, which should serve as a proxy of the counterfactual prediction measured on z_{test} (i.e. $f(z_{\text{test}}; \hat{\theta}_S)$). The *linear datamodeling score* (LDS) then defines the *goodness score* as the Spearman rank correlation (Spearman, 1904) between counterfactual predictions when the models are re-trained on different training subsets S and the scores computed by the attribution method. Formally, let $\{S_1, \ldots, S_m : S_j \subseteq D\}$ be m randomly sampled data subsets from the full training set D, each of size $\alpha \times n$ for some fixed $\alpha \in (0, 1)$. The *linear datamodeling score* (LDS) of τ for a specific example z_{test} is defined as:

$$\text{LDS}(\tau, z_{\text{test}}) \triangleq \boldsymbol{\rho}(\{f(z_{\text{test}}; \hat{\theta}_{S_j}) : j \in [m]\}, \{g_\tau(z_{\text{test}}, S_j; D) : j \in [m]\}),$$

where $\boldsymbol{\rho}$ is the Spearman rank correlation (Spearman, 1904).[18] Compared to LOO influence, only a constant number of training data subsets are required to compute an accurate LDS, regardless of the size of the training data set (typically only tens to hundreds of models are sufficient for the LDS to converge (Park et al., 2023)). Notable experiment settings with LDS are summarized in Table 15.

[17] If a data attribution method is additive, then it defines an attribution score that the overall influence of a group is the sum of the individual influence in the group.

[18] It is straightforward to extend the definition of LDS to a set of test examples (i.e. $D_{\text{eval}} = \{z_{\text{test}}^{(j)}\}$) by $\text{LDS}(\tau, D_{\text{eval}}) = \frac{1}{|D_{\text{eval}}|} \sum_{z_{\text{test}} \in D_{\text{eval}}} \text{LDS}(\tau, z_{\text{test}})$

Limitations. While being more computationally efficient than LOO influence, LDS still faces the following limitations.

- **Spearman rank correlation:** As discussed by Epifano et al. (2023), the Spearman rank correlation coefficient (Spearman, 1904), often used as a *goodness score* for evaluating data attribution, captures only monotonic relationships and ignores scale. Thus, it may yield high correlation even when the estimated and true influence values diverge significantly in magnitude, potentially masking estimation inaccuracies.
- **Assumption on additivity:** The assumption on the additivity of attribution scores can be violated and might hinder LDS from probing the *true* ability of attribution methods on predicting the behavior of re-trained models when they are trained with a significant amount of data points removed (Hu et al., 2024a).

Generative model settings. Park et al. (2023) evaluate their main method, TRAK, with LDS on mostly small-scale classification tasks (i.e. the parameter size of models is much smaller then modern generative models). The only generative setting discussed in the paper is conducted on a `mt5-small` model (Xue et al., 2021) ($\approx$ 300M parameters), which is much smaller then modern generative models. Even much cheaper than LOO influence, training hundreds of model variant is still computationally extensive for *large language models* (typically with tens to hundreds of billion parameters). Thus, it is more often for papers discussing *diffusion models* to evaluate their proposed methods using LDS. For example, these data attribution methods designed for diffusion models (Georgiev et al., 2023; Zheng et al., 2023; Brokman et al., 2024; Lu et al., 2025; Bae et al., 2024; Wang et al., 2024d) all include LDS as one evaluation protocol.

Table 15: Representative experiment settings with LDS.

Reference	Models	Datasets
Diffusion Models		
Georgiev et al. (2023)	DDPM (Ho et al., 2020b), LDM (Rombach et al., 2022)	CIFAR-10 (Krizhevsky, 2009), MS COCO (Lin et al., 2014)
Zheng et al. (2023)	DDPM (Ho et al., 2020b)	CIFAR-2/10 (Krizhevsky, 2009), ArtBench (Liao et al., 2022), CelebA (Liu et al., 2015)
Lin et al. (2025)	DDPM (Ho et al., 2020b), Stable Diffusion (Rombach et al., 2022)	CIFAR-2/10 (Krizhevsky, 2009), ArtBench (Liao et al., 2022)
Lu et al. (2025)	DDPM (Ho et al., 2020b), Stable Diffusion (Rombach et al., 2022)	CIFAR-20 (Krizhevsky, 2009), ArtBench (Liao et al., 2022), CelebA-HQ (Karras et al., 2018)

4.1.3 Steering Model Output Change

In contrast to the previous methods, another line of evaluation focuses on the general utility of attribution scores through data removal or addition experiments. Rather than comparing attribution scores to ground-truth outputs, these methods assess how model output changes when specific subsets of the training data are removed or added based on their attribution scores. The key hypothesis is that removing the most influential samples, which are those with the top-k highest (most positively attributed) scores, should degrade performance, while removing the least useful or harmful samples (those with the bottom-k scores) should improve it. This form of steering offers a practical way to validate the effectiveness of a data attribution method. For this type of evaluation method, the definition of *goodness score* varies between different scenarios, which includes but not limits to the magnitude of the prediction logit or loss change (Yeh et al., 2022; Wang et al., 2024h,h; Guo et al., 2021; Barshan et al., 2020; Koh et al., 2024; Lu et al., 2025), accuracy drop (Park et al., 2023; Fotouhi et al., 2024; Wu et al., 2023b; Lin et al., 2024c; Wang et al., 2024g), and change of FID score (Georgiev et al., 2023; Lin et al., 2024a). Representative experiment setups that inspect model output are summarized in Table 16.

Brittleness. Brittleness (Ilyas et al., 2022; Moitra and Rohatgi, 2023), which is originally defined as *finite-sample stability* by Broderick et al. (2020) and Moitra and Rohatgi (2023), is another important evaluation metric probing the prediction change of the model trained with a valuable set of data removed.[19] Formally, given D as the full training

[19]We note that *brittleness* was originally defined in the context of traditional classification problems. Accordingly, we adopt the same setting for its definition.

dataset and z_{test} a test example, we define the *data support* of z_{test}, denoted as SUPPORT(z_{test}), as the smallest subset $S \subset D$ such that the *counterfactual* classifier trained on $D \setminus S$ *mis-classifies* z_{test}:

$$\text{SUPPORT}(z_{\text{test}}) = \min_{S \subset D} \{|S|, \text{ where } f(z_{\text{test}}; \hat{\theta}_{D \setminus S}) \neq y_{z_{\text{test}}}\},$$

where $f(z_{\text{test}}; \hat{\theta}_{D \setminus S})$ is the (optimal) classifier trained on $D - S$ and $y_{z_{\text{test}}}$ is the ground-truth label of z. Specifically, a test example z_{test} is considered *brittle* if SUPPORT(z_{test}) is small, i.e., removing only a few training points can significantly change the model's prediction on z_{test}. Notably, Ilyas et al. (2022) proposes an algorithm that is suitable for any data attribution method to approximate *brittleness* utilizing its corresponding attribution scores for a guided search on optimal S. Several later papers (Park et al., 2023; Choe et al., 2024) evaluate their proposed data attribution method via the same algorithm design to compute *brittleness*.

Limitations. When measuring brittleness, previous papers (Ilyas et al., 2022; Park et al., 2023) typically first apply a specific algorithm, which utilizes the assumption of additivity of attribution scores (i.e. group influence can be approximated by the sum of individual influence) to obtain the *data support* for each test sample. This assumption is not necessarily true and thus contributes to a mismatch gap between the exact utility of data attribution method and the brittleness output by the algorithm (Hu et al., 2024a).

Generative model settings. While brittleness is originally used to probe classification tasks (Ilyas et al., 2022), it can be extended to generative scenarios by modifying the definition of the target function $f(z; \theta)$. For example, Park et al. (2023) and Choe et al. (2024) compute the effectiveness of their proposed methods with the generative language modeling setting, where for any text query in the training corpus, the classification target can be defined as the next token associated with the ending token of the query.

Table 16: Representative experiment settings with model output change.

Reference	Models	Datasets	Metrics	Task
Language Models				
Bae et al. (2024)	GPT-2 XL (Radford et al., 2019)	Wikitext (Merity et al., 2017)	Perplexity	Brittleness
Wang et al. (2025a)	GPT-2 small (Radford et al., 2019)	Pile (Gao et al., 2020)	loss change	Counterfactual
Diffusion Models				
Georgiev et al. (2023)	DDPM (Ho et al., 2020b), LDM (Rombach et al., 2022)	CIFAR-10 (Krizhevsky, 2009), MS COCO (Lin et al., 2014)	FID change ℓ_2-distance change	Brittleness Counterfactual
Zheng et al. (2023)	DDPM (Ho et al., 2020b)	CIFAR-2/10 (Krizhevsky, 2009), ArtBench (Liao et al., 2022), CelebA (Liu et al., 2015)	ℓ_2-distance change CLIP-similarity	Counterfactual
Wang et al. (2024h)	LDM (Rombach et al., 2022)	MS COCO (Lin et al., 2014)	loss/ℓ_2-distance change, CLIP-similarity	Counterfactual

4.2 Effectiveness in Downstream Tasks

Overview. Application-based evaluation of data attribution methods assesses their practical utility in real-world scenarios. Key application domains include *data selection* (Section 3.1), *fact tracing* (Section 3.2) and *adversarial attacks* and *defenses* (Section 3.3). We focus on these areas for their heightened relevance in the era of generative AI. For each downstream application, we further discuss the associated evaluation metrics, common limitations as well as caveats in applying them to generative modeling contexts.

4.2.1 Data Selection

In this evaluation setting, attribution methods are applied to edit the training dataset and their effectiveness is judged by the change in downstream model performance. The underlying assumption is that a well-calibrated attribution method assigns high scores to genuinely useful examples and low scores to detrimental ones. Accordingly, retaining top-ranked data or removing low-quality entries should yield performance improvements, while the inverse should lead to degradation. The following paragraphs examine common applications of this setting: *Selection and Filtering*, which focuses on retaining or discarding samples based on their scores, and *Mislabeled Data Detection*, which targets the identification and removal of incorrectly annotated examples. Notable experiment settings are summarized in Table 17.

Selection and filtering. Whether aiming to retain only the most valuable samples or remove those harmful ones, data attribution scores are ultimately judged by their impact on downstream performance. As mentioned in Section 3.1.1, a common pipeline is to utilize the score to retain, discard, or re-weight training samples, and then re-train or fine-tune the model on the edited dataset. The updated model is evaluated on a target dataset, with metrics such as accuracy and mean log-probability on a holdout set being standard choices (Engstrom et al., 2024; Xia et al., 2024; Zhou et al., 2024; Wang et al., 2024e; Zhou et al., 2024; Yu et al., 2024; Wang et al., 2024e, 2025b, 2023c; Kang et al., 2023; He et al., 2024; Chhabra et al., 2024; Zhou et al., 2024; Wang et al., 2024e).

When the goal is to filter out low-quality or toxic data (Wang et al., 2024b; Bejan et al., 2023; Schioppa et al., 2022), attribution scores guide the identification of such samples. In some cases, their influence is explicitly "unlearned" by altering the model to forget them (Wang et al., 2024h; Lin et al., 2024a; Ye et al., 2024). Here too, effectiveness is measured by improvements in downstream metrics, supplemented by task-specific measures such as F1 or exact match score in question answering tasks (Bejan et al., 2023; Xie et al., 2024b; Teso et al., 2021).

Mislabeled/noisy data detection. As another important application, *mislabeled/noisy data detection* aims to identify incorrect labels/noisy features in training datasets. In earlier works, mislabeled/noisy data is curated manually and has been a standard task to for evaluation (Koh and Liang, 2017; Khanna et al., 2019; Pruthi et al., 2020; Kim et al., 2023; Kwon et al., 2024; Zhou et al., 2024; Ko et al., 2024; Bejan et al., 2023; Schioppa et al., 2022; Just et al., 2023).

To detect mislabeled/noisy data, different indicators are used to rank the training samples. For example, gradient-based methods (Koh and Liang, 2017; Pruthi et al., 2020; Park et al., 2023) typically compute *self-influence* by imaging a test set with the training data point in interest within it and computing the attribution scores between them. In contrast, shapley-based methods (Ghorbani and Zou, 2019; Li and Yu, 2023; Wang and Jia, 2023a) do not require specifying the imaginary test set. They assess the contribution of each training example by considering its marginal impact directly. A high self-influence score indicates a mislabeled or noisy data point, suggesting that the remaining training data cannot adequately compensate for its absence.

To evaluate the effectiveness of data attribution methods in identifying such points, most studies compute false positive and true positive rates under varying thresholds (i.e., number of retrieved samples). Common evaluation metrics include Area Under the Curve (AUC) (Pruthi et al., 2020; Schioppa et al., 2022; Wang et al., 2025a; Just et al., 2023; Ko et al., 2024; Kwon et al., 2024), Average Precision (AP) (Schioppa et al., 2022), and F1 score (Cai, 2024). For this evaluation protocol, the experiment setting is primarily designed for traditional classification settings rather than generative ones. Therefore, we omit a detailed coverage in Table 17.

Limitations. Despite the practical appeal, several important limitations should be noted:

- **Dependence on high-quality validation set:** Data selection and cleaning rely on the existence of a reliable validation dataset to serve as a proxy for measuring attribution effectiveness. However, constructing such a dataset is often non-trivial or infeasible in many real-world settings.
- **Limited generality of mislabeled data detection.** Although mislabeled data detection is a widely studied task, spanning gradient-based (Koh and Liang, 2017; Pruthi et al., 2020; Schioppa et al., 2022), data Shapley-based (Wang et al., 2024d; Kwon and Zou, 2022; Wang and Jia, 2023a; Li and Yu, 2023; Kwon and Zou, 2023), and more recent approaches (Zhou et al., 2024; Ko et al., 2024; Just et al., 2023; Bejan et al., 2023), its relevance to evaluating data attribution methods remains questionable. These methods often rely on *self-influence* scores, ignoring the relative ranking across the dataset, which contradicts key principles of attribution evaluation.

Generative model settings. We summarize how the *data-selection evaluation* protocol is specifically instantiated for large language models and diffusion models.

- **Language models:** Due to the scale of modern LLMs, the gold-standard of full pre-training on a selected subset is practically impossible. As a result, most studies resort to continued pre-training of a fixed checkpoint or fine-tuning on a pruned corpus. This introduces a systematic mismatch: performance gains (or losses) observed under light fine-tuning may not extrapolate to those achievable via full-scale training, undermining the fidelity of the evaluation. Moreover, if the chosen performance oracle fails to capture the intended LLM capability, attribution-guided selection can lead to misleading conclusions about a TDA method's effectiveness.
- **Diffusion models:** In vision-generative settings, stochastic sampling and guidance-scale sensitivity introduce substantial noise into fidelity and alignment metrics. Thus, attribution scores can misrepresent an image's true utility and produce unstable evaluation signals. These issues are amplified in text-conditioned diffusion: image–caption pairs form joint representations, meaning that pruning based on single-modality influence can degrade the complementary modality's fidelity results toward high-frequency patterns.

Table 17: Representative experiment settings with data selection.

Reference	Models	Datasets	Metrics	Task
Language Models				
Wang et al. (2025b)	GPT-2 small (Radford et al., 2019) Pythia-410M (Biderman et al., 2023)	Wikitext (Merity et al., 2017), Pile (Gao et al., 2020)	Accuracy	Pre-train
Yu et al. (2024)	Pythia-410M/1B (Gao et al., 2020)	C4 (Raffel et al., 2020)	Accuracy	Pre-train
Thakkar et al. (2023)	mT5-base/large (Xue et al., 2021)	C4 (Raffel et al., 2020)	F1, Accuracy	Pre-train
Jiao et al. (2025)	Pythia-1B (Biderman et al., 2023), Llama-3.2-3B (Dubey et al., 2024)	Alpaca (Taori et al., 2023), K-Means-100 (Li et al., 2024c)	Accuracy	Fine-tune
Wang et al. (2024e)	Llama-2-7B (Touvron et al., 2023b), Llama-3.2-3B (Dubey et al., 2024), Mistral-7B (Jiang et al., 2023), GPT-2 small (Radford et al., 2019)	LESS data (Xia et al., 2024), Alpaca (Taori et al., 2023), OpenWebText (Gokaslan and Cohen, 2019)	Accuracy, Perplexity	Fine-tune
Xia et al. (2024)	Llama-2-7B/13B (Touvron et al., 2023b), Mistral-7B (Jiang et al., 2023)	MMLU (Hendrycks et al., 2021), TyDi QA (Clark et al., 2020), BIG-Bench Hard (Suzgun et al., 2023)	Accuracy	Fine-tune
San Joaquin et al. (2024)	Mistral-7B (Jiang et al., 2023), Gemma-2B (Team et al., 2024)	MMLU (Hendrycks et al., 2021), Ultrachat (Ding et al., 2023)	Accuracy, Perplexity	Fine-tune
Vision-Language Models				
Zhou et al. (2024)	Llava (Liu et al., 2023)	Prismatic VLMs instructions (Karamcheti et al., 2024)	Accuracy	Pre-train
Liu et al. (2024b)	Llava-1.5-7B/13B (Liu et al., 2024a)	LLaVA-1.5 instructions (Liu et al., 2024a), SVIT-Mix (Zhao et al., 2023), Mini-Gemini (Li et al., 2024b)	Accuracy	Fine-tune
Diffusion Models				
Xie et al. (2024a)	DDIM (Song et al., 2021a)	MNIST (LeCun et al., 1998), CIFAR-10 (Krizhevsky, 2009)	Precision	Mislabeled

4.2.2 Fact Tracing and Concept Probing

For these two applications, researchers primarily assess whether a data attribution method can successfully retrieve a predefined set of training examples that represent specific *facts* or *concepts*. The goal is to determine whether high- or low-scoring training samples align with known positively or negatively influential examples. These may include fact-providing instances or erroneous and harmful samples in a poisoned dataset. Representative experiment settings are summarized in Table 18.

Fact tracing. The goal of fact tracing (Park et al., 2023; Akyürek et al., 2022; Ko et al., 2024) is to evaluate whether a data attribution method can accurately identify the "ground-truth" training examples that served as factual sources for a given test prediction. A closely related task is *hallucination tracing*, which aims to attribute targeted hallucinations in model outputs back to perturbed or misleading training examples (Wu et al., 2024b; Lin et al., 2024b). Fact tracing is typically framed as a retrieval task. Its effectiveness is measured using standard ranking metrics such as Mean Reciprocal Rank (MRR) (Park et al., 2023; Liu and Yang, 2024; Chang et al., 2025) and Recall@k (Akyürek et al., 2022; Chang et al., 2025; Wu et al., 2024b). MRR evaluates the rank position of the first correctly identified ground-truth fact, assigning higher scores when relevant examples appear closer to the top while Recall@k measures the proportion of test instances for which the true fact appears among the top-k most attributed training examples.

Tail-patch. In contrast to traditional retrieval-based metrics, which often assume that the top-k retrieved samples contain the target fact, Chang et al. (2025) propose *tail-patch* as a more direct evaluation strategy that incorporates model influence. Specifically, this evaluation method takes one additional training step using a single top-ranked training example (retrieved based on attribution scores) from the final model checkpoint, and measures the change in the predicted probability of the target test sample. This directly quantifies the causal impact of each retrieved example on the model's behavior.

Attribution by customization. Wang et al. (2023b) introduce a novel evaluation framework called *Attribution by Customization* (AbC) for text-to-image models. AbC involves fine-tuning a pre-trained diffusion model on a single or a set of exemplar images. It extends the concept of fact tracing by treating exemplar images as ground-truth facts and evaluating whether the generated outputs can be correctly attributed back to these sources. Similar to other retrieval-based evaluations, AbC employs metrics such as Recall@k and mean average precision (mAP) to assess attribution performance. Wang et al. (2024h) and Brokman et al. (2024) also follow this setting to evaluate their attribution methods on diffusion models.

Concept probing. In contrast to fact tracing, which is relatively straightforward to define *ground-truth* labels for training samples, concept probing presents additional challenges due to the vague and context-dependent nature of what constitutes a "concept." The definition of a concept often varies depending on the task and use case. For language modeling, one notable scenario is in the context of multilingual language models where the goal is to investigate how training data in one language influences performance in another. Here, *concept* of a specific training data can be explicitly defined as its corresponding language (Choenni et al., 2023, 2024a). The evaluation metric is termed *Top-k Contribution*, which refers to the average percentage of training samples from each fine-tuning language that contributed to the top-k training samples for a test language. For diffusion models, Brokman et al. (2024) uses the *CustomConcepts101* dataset (Kumari et al., 2023), which is a benchmark dataset encompassing various concepts and their corresponding textual prompts. Similarly, Kwon et al. (2024); Brokman et al. (2024) build customized text-to-image datasets with *ground-truth* concepts to evaluate their proposed methods.

Limitations. For fact tracing applications, the main limitation lies in the acquisition of high-quality ground-truth fact defined in the training data. This is also reflected by the scarcity of the available dataset used to test the attribution methods dedicated to solve these two applications. For example, FTRACE-TREx (Akyürek et al., 2022) and its variant (Akyürek et al., 2022; Chang et al., 2025) are the only golden-standard datasets for the fact tracing evaluation (Ko et al., 2024; Wu et al., 2024b; Park et al., 2023).

Generative model settings. We describe how fact tracing and concept probing are adapted and evaluated in two major classes of generative models, namely language models and diffusion models, as follows.

- **Language models:** In this setting, language models are fine-tuned on a fact tracing dataset consisting of abstracts and their corresponding facts, using a masked language modeling (MLM) objective. Since MLM involves predicting masked tokens from a fixed vocabulary, the task naturally aligns with a v-way classification problem, where v is the vocabulary size. As discussed in TRAK (Park et al., 2023), this framing enables the application of data attribution methods originally developed for multi-class classification tasks.
- **Diffusion models:** For image generation models, fact tracing and concept probing are often referred to as *image tracing*, where the goal is to identify specific training images that contribute to the generation of a given output. These images may share common *styles*, *subjects*, or *categories*. For instance, Xie et al. (2024a) construct a synthetic dataset by mixing MNIST (LeCun et al., 1998) and CIFAR-10 (Krizhevsky et al., 2010), and train a diffusion model on this combined set. For each test sample, they compute the proportion of correctly attributed training examples among the top-k influential samples as the evaluation metric.

Table 18: Representative experiment settings with fact tracing and concept probing.

Reference	Models	Datasets	Metrics	Task
Language Models				
Akyürek et al. (2022) Ko et al. (2024)	mT5-base (Xue et al., 2021)	FTRACE-TREx (Akyürek et al., 2022)	MRR, Recall, Precision	Fact Tracing
Park et al. (2023) Liu and Yang (2024)	mT5-small (Xue et al., 2021)	FTRACE-TREx (Akyürek et al., 2022)	MRR	Fact Tracing
Lin et al. (2024b)	Llama-2-7B/70B (Touvron et al., 2023b)	Alpaca Taori et al. (2023)	Precision	Fact Tracing
Chang et al. (2025)	PALM-154M/1B/8B (Chowdhery et al., 2023)	T-REx (Elsahar et al., 2018), C4 (Raffel et al., 2020)	Tail-Patch, MRR, Recall	Fact Tracing
Choenni et al. (2024a)	mT5-small/base/large (Xue et al., 2021)	PBC (Christodouloupoulos and Steedman, 2015), Tanzil (Tiedemann, 2012)	Top-100 Contribution	Concept Probing
Choenni et al. (2023)	XLM-R Base (Xue et al., 2021)	XNLI (Conneau et al., 2018), PAWS-X (Yang et al., 2019), MARC (Keung et al., 2020)	Top-100 Contribution	Concept Probing
Choenni et al. (2024b)	XLM-R Base (Xue et al., 2021)	XNLI (Conneau et al., 2018), PAWS-X (Yang et al., 2019), MARC (Keung et al., 2020)	Accuracy	Concept Probing
Diffusion Models				
Xie et al. (2024a)	DDIM (Song et al., 2021a)	MNIST (LeCun et al., 1998), CIFAR-10 (Krizhevsky, 2009)	Precision	Fact Tracing
Brokman et al. (2024)	Stable Diffusion (Rombach et al., 2022)	CustomConcepts101 (Kumari et al., 2023), Artchive (Harden, Accessed 2025)	Recall, Precision, Spearman Corr.	Concept Probing, AbC
Wang et al. (2023b) Wang et al. (2024h)	DINO (Caron et al., 2021), CLIP (Radford et al., 2021)	LAION (Schuhmann et al., 2022)	Recall, mAP	AbC

4.2.3 Adversarial Attack and Defense

Evaluation of adversarial attack/defense application involves deliberately perturbing or poisoning the training data, and measuring how effectively data attribution methods can identify these (negatively) influential samples. For instance, Lin et al. (2024b); Xie et al. (2024b) construct poisoned dataset and apply attribution methods to locate the injected harmful data samples. Detection performance is typically quantified using metrics like Area Under the Precision-Recall Curve (AUPRC) and Area Under the Receiver Operator Characteristic Curve (AUROC) (Xie et al., 2024b; Lin et al., 2024b). On the defense side, evaluation methods can also assess how attribution methods help mitigate harmful impacts by guiding model fine-tuning or data selection processes. For example, studies (Choe et al., 2024; Barshan et al., 2020) study model responses to fine-tuning on harmful data and evaluate improvements using safety-oriented metrics, such as Attack Success Rate (ASR) and output loss reduction. The notable experiment settings with adversarial attack/defense are summarized in Table 19.

Table 19: Representative experiment settings with adversarial attack/defense.

Reference	Models	Datasets	Metrics	Task
Language Models				
He et al. (2024)	Llama-2-7B/13B-Chat (Touvron et al., 2023b), Llama-3-8B-Chat (Dubey et al., 2024), Gemma-7B-Instruct (Team et al., 2024)	Alpaca (Taori et al., 2023), Dolly (Conover et al., 2023)	GPT score, GPT ASR, ASR increase	Attack
Lin et al. (2024b)	Llama-2-7B/70B (Touvron et al., 2023a)	Poisoned Alpaca (Taori et al., 2023)	AUPRC, AUROC	Attack
He et al. (2024)	Llama-2-7B-Chat (Touvron et al., 2023b)	Alpaca (Taori et al., 2023), Dolly (Conover et al., 2023)	ASR reduction	Defense
Xie et al. (2024b)	Llama-2-7B (Touvron et al., 2023a)	ToxicChat (Lin et al., 2023), XSTest (Röttger et al., 2024)	AUPRC, F1, Precision, Recall	Defense

4.3 Computational Costs

Parallel to the evaluation on the effectiveness of data attribution methods, time/memory efficiency metrics such as *runtime* (Lin et al., 2024b; Wang et al., 2024h; Koh et al., 2024; Yu et al., 2024; Wang et al., 2024e; Brokman et al., 2024; Kwon et al., 2024; Ko et al., 2024; Schioppa et al., 2022; Yang et al., 2025; Kwon and Zou, 2023; Wang et al., 2024g), *throughput* (Wang et al., 2024e, 2025b), *memory storage* (Wang et al., 2025b; Lin et al., 2024b; Xia et al., 2024) and *peak GPU memory consumption* (Koh et al., 2024; Deng et al., 2024a; Wang et al., 2025b) are all commonly appeared in the data attribution literature.

4.4 Other Evaluation Criteria

Loss curve fitness. For simulator methods (Guu et al., 2023; Chai et al., 2024), the evaluation metrics test the ability to predict the entire training loss trajectory and the final training loss. Thus, the mean squared error (MSE) aggregated throughout the loss trajectory and the correlation coefficient between the true final-step training loss and the predicted one are the main metric (Guu et al., 2023; Chai et al., 2024; Yoon et al., 2020).

4.5 Suggestions for Future Evaluations

To address existing limitations of evaluation paradigms for data attribution methods, the following directions can be considered when proposing new evaluation protocols in the future:

4.5.1 Beyond Multi-model Training

According the the previous section, data attribution methods across various models trained on counterfactual datasets can provide a more comprehensive understanding of data influence. However, training multiple models is computationally expensive, which is involved in the computation of LOO influence and LDS.

Recommendation. To reduce the computational burden of multi-model training while still capturing the ability of data attribution to predict counterfactuals, future evaluation protocols should explore scalable approximations to evaluate large generative models. For example, researchers can investigate surrogate modeling to reduce model size or ensemble approximation techniques to cut down number of re-trained models needed. These approaches could help preserve the quality of attribution assessments while significantly lowering computational costs, especially when dealing with large generative language and diffusion models.

4.5.2 Attribution Scores as Distributions

For the first time, Nguyen et al. (2023) introduces a Bayesian formulation of training data attribution. Rather than treating a trained deep model as a fixed entity, the paper embraces the inherent randomness during model training by considering the resulting model as a sample from a posterior distribution. The paper also offers the following key insights: (i) *noise dominance*: the variance in the data attribution scores can be larger than their average influences and thus mean estimates might be misleading, (ii) *score reliability*: for each pair of training-test data, only those with substantial signal out-weighting the noise can be deemed reliable, and (iii) *inaccurate evaluation*: stand metrics like Pearson or Spearman correlation coefficients (Pearson, 1895; Spearman, 1904) may fail when the uncertainty is ignored during the data attribution score comparison.

Recommendation. Based on the observation aforementioned, we can conclude that analyzing the standard deviation and other statistical properties of the data attribution scores can reveal the consistency and reliability of the attributions. We encourage the development of evaluation methods that account for the inherent variability in model training and data sampling processes, together with more nuanced evaluation metrics that are independent of the internal variances.

4.5.3 Benchmark Suite with Standardized Settings

Establishing a comprehensive benchmark suite that includes a series of downstream applications with standardized settings, including fixed combinations of models, datasets, and evaluation metrics, would facilitate consistent and fair comparisons of data attribution methods. Recent efforts, such as the development of the `dattri` library developed by Deng et al. (2025), have aimed to provide unified APIs and benchmark frameworks to streamline the evaluation process. Moreover, the suite must cover the full spectrum of generative model classes, including relatively modern large language models and diffusion models, so that attribution methods can be evaluated under consistent, real-world settings. Such a standardized, transparent framework will accelerate methodological progress and build confidence in attribution techniques across diverse use cases.

Recommendation. For future TDA research, we recommend including LDS evaluation alongside downstream applications such as data selection experiments (fine-tuning on or without selected influential subsets) and explicit reporting of computational cost (GPU-hours, memory storage and peak GPU memory consumption).

5 The Scope of the Survey

Scope. This survey focuses on data attribution in modern machine learning, with a particulary emphasis on generative AI. Prior to this work, there was only one survey dedicated to the topic of data attribution (Hammoudeh and Lowd, 2024)[20], yet their focus was on conventional machine learning models and did not cover recent state-of-the-art development. We also note that while the concept of data attribution has historical roots in fields such as statistics and econometrics (e.g., robust statistics (Huber and Ronchetti, 2009), regression diagnostics (Belsley et al., 2005), and game-theoretic Shapley values (Shapley, 1953)), these antecedents fall outside the scope of this survey.

Our coverage of data attribution methods (Section 2) is guided by two key criteria: broad representation of major data attribution paradigms and their (potential) applicability to generative AI models. Notably, although weighted marginal contribution methods (Section 2.3) are often computationally infeasible for generative AI due to repeated retraining, we provide a comprehensive overview given their central role in the data attribution literature and the lack of systematic treatment in existing surveys (Hammoudeh and Lowd, 2024). Our discussion of applications (Section 3) centers on generative AI but also includes scenarios in conventional machine learning (e.g., adversarial attack and defense in Section 3.3) where applications in generative AI are currently limited but transferable in principle.

The systematic literature search for this version of survey paper was conducted in March 2025[21] and focused on influential and peer-reviewed studies related to data attribution with a particular focus on generative AI. The process began with approximately 40 seed papers selected based on their topical relevance. We then employed a snowball sampling strategy to expand the literature list by including papers cited by or citing the initial seed papers. To ensure quality, we filter the papers with proxy criteria based on their publication status and citation count: papers were included only if they were published in peer-reviewed journals or conferences, or if they were well-cited, defined as having a citation count greater than or equal to the number of months since the paper's first preprint became available online.

[20] They refer the field as "training data influence analysis", while the term "data attribution" has become more standard recently.

[21] We have planned an updated literature search to be included in the next version of this survey paper.

Related topics. We briefly discuss the distinction between data attribution and some closely related topics to better clarify the scope of this survey paper.

1. **Data monetization.** *Data monetization*, sometimes also referred to as "data valuation" (Fleckenstein et al., 2023), focuses on deriving economic benefits from data (Najjar and Kettinger, 2013; Faroukhi et al., 2020). In contrast, data valuation or data attribution as named in this survey paper typically emphasizes the contribution of individual data points or subsets to a model's behavior with various applications, rather than directly assigning a market price. While these concepts are related, such data attribution scores may serve as a prerequisite for monetization but do not necessarily involve commercialization, we have some discussion about this in Section 3.4.
2. **Membership inference attacks.** *Membership inference attack* (MIA) aims to determine whether a specific data sample was included in the training set of a given model (Shokri et al., 2017; Salem et al., 2019; Parisot et al., 2021; Hu et al., 2022b). Such attacks are widely used to estimate the privacy risks of machine learning systems, particularly in sensitive application domains such as medical records or personal user data (Shokri et al., 2017; Truex et al., 2019). While both MIA and data attribution examine the relationship between a model and its training data, their objectives and assumptions differ. MIA focuses on a binary classification problem—inferring whether a given sample was included in the training set. In contrast, data attribution methods aim to quantify the influence of individual training points on model predictions (Koh and Liang, 2017), typically under the assumption that the full training set is known or accessible. Despite these differences, both approaches leverage signals from models (e.g., prediction confidence, gradients, or loss) and share methodological overlaps, which are discussed in Section 3.3.4.
3. **Machine unlearning.** *Machine unlearning*, which focuses on removing the influence of specific training data from a trained model, often to comply with data removal requests such as those mandated by the "right to be forgotten" (Cao and Yang, 2015; Ginart et al., 2019; Neel et al., 2021). Unlike data attribution, which typically does not modify a machine learning model, machine unlearning aims to *reverse* the model so it behaves as if the target data sample had never been used for training. Existing approaches range from exact retraining to approximate strategies like certified removal guarantees (Golatkar et al., 2020; Bourtoule et al., 2021). Both areas benefit from a sample-level understanding of data and model interactions, and in recent years, the advances of data attribution methods and machine unlearning methods are shown to improve each other, which is discussed in Section 2.6.1.
4. **Differential privacy.** *Differential privacy* (DP) (Dwork and Roth, 2014; Abadi et al., 2016) is a mathematical framework that limits what an adversary can infer about any single training example from an algorithm's output, and has been widely adopted in deep learning (De et al., 2022; Yu et al., 2022; Hu et al., 2024b). Unlike data attribution, which seeks to quantify individual data influence, DP aims to cap it through randomization. Despite this difference in goals, both operate at the level of individual samples and thus share methodological similarities. For example, per-example gradients are central to both: in data attribution, they estimate sample influence; in DP, they are clipped to bound it. The ghost dot-product technique in Section 2.4.1 is one such crossover, adapted from the ghost clipping method (Li et al., 2022) originally developed for DP.

6 The Future of Data Attribution in GenAI

Looking ahead, this section presents a research agenda for data attribution in the era of Generative AI, outlining key directions intended to guide future work. We delineate our vision for data attribution within this transformative landscape, identifying the most impactful challenges and opportunities ahead. These points are designed to serve as both a practical blueprint for concrete research endeavors and a northstar to orient the field towards realizing its full potential.

1. **Scale as a central consideration in data attribution for GenAI.** The defining characteristic of state-of-the-art GenAI is its operation at scales previously unimaginable—billions of parameters trained on petabytes of data. Scale is not merely a technical hurdle for data attribution methods; it fundamentally shapes the problem itself, influencing model behavior, data interactions, and computational feasibility. Therefore, a **scale-aware mindset** must permeate all research in GenAI attribution. This imperative translates into several interconnected research thrusts, including: (1) Development of scalable attribution methods (Wang et al., 2025a; Chang et al., 2025); (2) Studying the scaling properties of data influence (Covert et al., 2024); and (3) Leveraging cross-scale insights (especially small-to-large) (Kang et al., 2024; Yang et al., 2024; Khaddaj et al., 2025).
2. **Data attribution for GenAI's diverse data & paradigms.** GenAI models are characterized by the heterogeneity of their training data—spanning diverse modalities (text, image, video), sources (massive

web scrapes, curated datasets, synthetic data), and types (raw data, human preferences). Furthermore, their development involves complex multi-stage processes incorporating various learning paradigms (pre-training, supervised fine-tuning, RLHF). Data attribution must evolve to handle this multifaceted landscape. This requires developing attribution strategies capable of: (1) Handling diverse data modalities; (2) Adapting to more sophisticated learning paradigms, such as RL and RLHF (Hu et al., 2025); (3) Understanding cumulative impact across multi-stage training (e.g., the development of frontier LLMs is a continual effort, and often times we want to answer the following question: "*how much value does this new data point / data source add to the existing model?*"); and (4) Accounting for the usage of synthetic data.

3. **Data attribution as a foundational tool for interpreting and governing responsible GenAI.** Data attribution serves as a critical lens for peering into the opaque nature of large generative models, providing essential interpretability and debugging capabilities. While data attribution has a history in traditional machine learning, GenAI's unique complexity, emergent behaviors, and challenging failure modes significantly elevate its importance. It provides actionable insights into key concerns for responsible AI, including identifying data sources of bias, uncovering potential privacy leaks through memorization, tracing the origins of safety failures, and understanding intellectual property issues related to training data (e.g., copyright violations). By establishing a crucial link between model behavior and its training data, data attribution moves beyond being a purely technical tool to become an essential pillar for building and governing responsible AI systems, providing the transparency needed for ethical deployment and regulatory compliance.
4. **Shift towards proactive and dynamic attribution.** Moving data attribution from a purely *post-hoc* analysis to a more proactive and dynamic process integrated into the GenAI development lifecycle is crucial for efficiency, control, and continuous improvement. We encourage the field to explore methods that perform attribution during training to guide data sampling, weighting, or filtering in real-time. This enables early problem detection and more efficient use of massive datasets Coalson et al. (2025).
5. **Establish rigorous evaluation and standardized benchmarks for GenAI attribution.** To measure algorithmic advancement and ensure the practical utility of data attribution methods for GenAI, the field must move towards more application-oriented, challenging, and rigorous evaluation. This involves several key steps: (1) Quantifying the direct impact and effectiveness of attribution methods on crucial downstream GenAI applications, shifting focus from correlation analysis based on LOO and LDS (which can sometimes be misleading or easily "fooled"); (2) Establishing application-oriented benchmarks that capture the complexity, scale, and unique data/model characteristics of GenAI, going beyond simple tasks like mislabeled data detection; and (3) Standardizing evaluation protocols, defining clear and reproducible procedures within these benchmarks (e.g., use ensembles or multiple runs to mitigate noise in the evaluation process).
6. **Synergy between data attribution and other interpretability methods.** Combining insights from data attribution (understanding data-model relationships) with other interpretability techniques such as mechanistic interpretability (Elhage et al., 2021; Sharkey et al., 2025) (understanding model internals) offers a powerful path towards achieving truly comprehensive understanding, control, and improved development of GenAI models (Zhang et al., 2025c; Cohen-Wang et al., 2025; Hua et al., 2025).

References

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Naman Agarwal, Brian Bullins, and Elad Hazan. Second-order stochastic optimization for machine learning in linear time. *Journal of Machine Learning Research*, 18(116):1–40, 2017.

Ekin Akyürek, Tolga Bolukbasi, Frederick Liu, Binbin Xiong, Ian Tenney, Jacob Andreas, and Kelvin Guu. Towards tracing knowledge in language models back to the training data. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2429–2446, 2022.

Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, Colin Raffel, Shiyu Chang, Tatsunori Hashimoto, and William Yang Wang. A survey on data selection for language models. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=XfHWcNTSHp. Survey Certification.

Walter Edwin Arnoldi. The principle of minimized iterations in the solution of the matrix eigenvalue problem. *Quarterly of applied mathematics*, 9(1):17–29, 1951.

Sanjeev Arora, Simon Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International conference on machine learning*, pages 322–332. PMLR, 2019.

Imanol Arrieta-Ibarra, Leonard Goff, Diego Jiménez-Hernández, Jaron Lanier, and E Glen Weyl. Should we treat data as labor? moving beyond "free". In *AEA Papers and Proceedings*, volume 108, pages 38–42. American Economic Association 2014 Broadway, Suite 305, Nashville, TN 37203, 2018.

Lars Backstrom and Jure Leskovec. Supervised random walks: predicting and recommending links in social networks. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 635–644, 2011.

Juhan Bae, Nathan Ng, Alston Lo, Marzyeh Ghassemi, and Roger B Grosse. If influence functions are the answer, then what is the question? *Advances in Neural Information Processing Systems*, 35:17953–17967, 2022.

Juhan Bae, Wu Lin, Jonathan Lorraine, and Roger Baker Grosse. Training data attribution via approximate unrolling. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=3NaqGg92KZ.

John F Banzhaf III. Weighted voting doesn't work: A mathematical analysis. *Rutgers L. Rev.*, 19:317, 1964.

Elnaz Barshan, Marc-Etienne Brunet, and Gintare Karolina Dziugaite. Relatif: Identifying explanatory training samples via relative influence. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 1899–1909. PMLR, 26–28 Aug 2020. URL https://proceedings.mlr.press/v108/barshan20a.html.

MS Bartlett. Approximate confidence intervals. *Biometrika*, 40(1/2):12–19, 1953.

Samyadeep Basu, Xuchen You, and Soheil Feizi. On second-order group influence functions for black-box predictions. In *International Conference on Machine Learning*, pages 715–724. PMLR, 2020.

Samyadeep Basu, Phil Pope, and Soheil Feizi. Influence functions in deep learning are fragile. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=xHKVVHGDOEk.

Irina Bejan, Artem Sokolov, and Katja Filippova. Make every example count: On the stability and utility of self-influence for learning from noisy NLP datasets. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10107–10121, Singapore, December 2023. Association for Computational Linguistics.

David A Belsley, Edwin Kuh, and Roy E Welsch. *Regression diagnostics: Identifying influential data and sources of collinearity*. John Wiley & Sons, 2005.

Alberto Bernacchia, Mate Lengyel, and Guillaume Hennequin. Exact natural gradient in deep linear networks and its application to the nonlinear case. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/7f018eb7b301a66658931cb8a93fd6e8-Paper.pdf.

Federico Bianchi, Mirac Suzgun, Giuseppe Attanasio, Paul Rottger, Dan Jurafsky, Tatsunori Hashimoto, and James Zou. Safety-tuned llamas: Lessons from improving the safety of large language models that follow instructions. In *The Twelfth International Conference on Learning Representations*, 2024.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR, 2023.

Bernd Bohnet, Vinh Q. Tran, Pat Verga, Roee Aharoni, Daniel Andor, Livio Baldini Soares, Jacob Eisenstein, Kuzman Ganchev, Jonathan Herzig, Kai Hui, Tom Kwiatkowski, Ji Ma, Jianmo Ni, Tal Schuster, William W. Cohen, Michael Collins, Dipanjan Das, Donald Metzler, Slav Petrov, and Kellie Webster. Attributed question answering: Evaluation and modeling for attributed large language models. *arXiv preprint arXiv:2212.08037*, 2022.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE symposium on security and privacy (SP)*, pages 141–159. IEEE, 2021.

Tamara Broderick, Ryan Giordano, and Rachael Meager. An automatic finite-sample robustness metric: when can dropping a little data make a big difference? *arXiv preprint arXiv:2011.14999*, 2020.

Carla E Brodley and Mark A Friedl. Identifying mislabeled training data. *Journal of artificial intelligence research*, 11: 131–167, 1999.

Jonathan Brokman, Omer Hofman, Roman Vainshtein, Amit Giloni, Toshiya Shimizu, Inderjeet Singh, Oren Rachmil, Alon Zolfi, Asaf Shabtai, Yuki Unno, and Hisashi Kojima. Montrage: Monitoring training for attribution of generative diffusion models. In *European Conference on Computer Vision*, pages 1–17. Springer, 2024.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Ma teusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Marc-Etienne Brunet, Colleen Alkalay-Houlihan, Ashton Anderson, and Richard Zemel. Understanding the origins of bias in word embeddings. In *International conference on machine learning*, pages 803–811. PMLR, 2019.

Huaiguang Cai. Chg shapley: Efficient data valuation and selection towards trustworthy machine learning, 2024. URL `https://arxiv.org/abs/2406.11730`.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186, 2017.

Bochuan Cao, Yuanpu Cao, Lu Lin, and Jinghui Chen. Defending against alignment-breaking attacks via robustly aligned llm. In *62nd Annual Meeting of the Association for Computational Linguistics, ACL 2024*, pages 10542–10560. Association for Computational Linguistics (ACL), 2024a.

Yihan Cao, Yanbin Kang, Chi Wang, and Lichao Sun. Instruction mining: Instruction data selection for tuning large language models. In *First Conference on Language Modeling*, 2024b. URL `https://openreview.net/forum?id=wF6k0aWjAu`.

Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *2015 IEEE symposium on security and privacy*, pages 463–480. IEEE, 2015.

Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom B. Brown, Dawn Xiaodong Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650, 2021.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

Javier Castro, Daniel Gómez, and Juan Tejada. Polynomial calculation of the shapley value based on sampling. *Computers & operations research*, 36(5):1726–1730, 2009.

Yekun Chai, Qingyi Liu, Shuohuan Wang, Yu Sun, Qiwei Peng, and Hua Wu. On training data influence of gpt models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3126–3150, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi:10.18653/v1/2024.emnlp-main.183. URL `https://aclanthology.org/2024.emnlp-main.183/`.

Tyler A. Chang, Dheeraj Rajagopal, Tolga Bolukbasi, Lucas Dixon, and Ian Tenney. Scalable influence and fact tracing for large language model pretraining. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=gLa96FlWwn.

Guillaume Charpiat, Nicolas Girard, Loris Felardos, and Yuliya Tarabalka. Input similarity from the neural network perspective. *Advances in Neural Information Processing Systems*, 32, 2019.

Yuanyuan Chen, Boyang Li, Han Yu, Pengcheng Wu, and Chunyan Miao. Hydra: Hypergradient data relevance analysis for interpreting deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7081–7089, 2021.

Anshuman Chhabra, Bo Li, Jian Chen, Prasant Mohapatra, and Hongfu Liu. Outlier gradient analysis: Efficiently improving deep learning model performance via hessian-free influence functions. *ArXiv*, abs/2405.03869, 2024.

Sang Keun Choe, Hwijeen Ahn, Juhan Bae, Kewen Zhao, Minsoo Kang, Youngseog Chung, Adithya Pratapa, Willie Neiswanger, Emma Strubell, Teruko Mitamura, Jeff G. Schneider, Eduard H. Hovy, Roger B. Grosse, and Eric P. Xing. What is your data worth to gpt? llm-scale data valuation with influence functions. *arXiv preprint arXiv:2405.13954*, 2024.

Rochelle Choenni, Dan Garrette, and Ekaterina Shutova. How do languages influence each other? studying cross-lingual data sharing during lm fine-tuning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13244–13257, 2023.

Rochelle Choenni, Anne Lauscher, and Ekaterina Shutova. The echoes of multilinguality: Tracing cultural value shifts during language model fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15042–15058, 2024a.

Rochelle Choenni, Ekaterina Shutova, and Dan Garrette. Examining modularity in multilingual lms via language-specialized subnetworks. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 287–301, 2024b.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

Christos Christodouloupoulos and Mark Steedman. A massively parallel corpus: the bible in 100 languages. *Language resources and evaluation*, 49(2):375–395, 2015.

Yung-Sung Chuang, Benjamin Cohen-Wang, Zejiang Shen, Zhaofeng Wu, Hu Xu, Xi Victoria Lin, James R. Glass, Shang-Wen Li, and Wen tau Yih. Selfcite: Self-supervised alignment for context attribution in large language models. In *Forty-second International Conference on Machine Learning*, 2025.

Jonathan H Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. Tydi qa: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470, 2020.

Zachary Coalson, Juhan Bae, Nicholas Carlini, and Sanghyun Hong. If-guide: Influence function-guided detoxification of llms. *arXiv preprint arXiv:2506.01790*, 2025.

Gilad Cohen and Raja Giryes. Membership inference attack using self influence functions. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 4892–4901, 2024.

Benjamin Cohen-Wang, Harshay Shah, Kristian Georgiev, and Aleksander Madry. Contextcite: Attributing model generation to context. *Advances in Neural Information Processing Systems*, 37:95764–95807, 2024.

Benjamin Cohen-Wang, Yung-Sung Chuang, and Aleksander Madry. Learning to attribute with attention. *arXiv preprint arXiv:2504.13752*, 2025.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, 2018.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. Free dolly: Introducing the world's first truly open instruction-tuned llm, 2023. URL https://www.databricks.com/blog/2023/04/12/dolly-first-open-commercially-viable-instruction-tuned-llm.

R. Dennis Cook and Sanford Weisberg. *Residuals and Influence in Regression*. Monographs on Statistics and Applied Probability. Chapman and Hall, New York, 1982.

Ian Covert and Su-In Lee. Improving kernelshap: Practical shapley value estimation using linear regression. In *International Conference on Artificial Intelligence and Statistics*, pages 3457–3465. PMLR, 2021.

Ian Connick Covert, Wenlong Ji, Tatsunori Hashimoto, and James Zou. Scaling laws for the value of individual data points in machine learning. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=scSB9RynSd.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. *arXiv preprint arXiv:2104.08696*, 2021.

Zheng Dai and David K Gifford. Training data attribution for diffusion models. *arXiv preprint arXiv:2306.02174*, 2023.

Soham De, Leonard Berrada, Jamie Hayes, Samuel L Smith, and Borja Balle. Unlocking high-accuracy differentially private image classification through scale. *arXiv preprint arXiv:2204.13650*, 2022.

Michiel Debruyne, Mia Hubert, and Johan AK Suykens. Model selection in kernel based regression using the influence function. *Journal of machine learning research*, 9:2377–2400, 2008.

Junwei Deng, Ting-Wei Li, Shichang Zhang, and Jiaqi Ma. Efficient ensembles improve training data attribution. *arXiv preprint arXiv:2405.17293*, 2024a.

Junwei Deng, Shiyuan Zhang, and Jiaqi Ma. Computational copyright: Towards a royalty model for music generative ai, 2024b. URL https://arxiv.org/abs/2312.06646.

Junwei Deng, Ting-Wei Li, Shiyuan Zhang, Shixuan Liu, Yijun Pan, Hao Huang, Xinhe Wang, Pingbang Hu, Xingjian Zhang, and Jiaqi Ma. dattri: A library for efficient data attribution. *Advances in Neural Information Processing Systems*, 37:136763–136781, 2025.

Xiaotie Deng and Christos H Papadimitriou. On the complexity of cooperative solution concepts. *Mathematics of operations research*, 19(2):257–266, 1994.

Shizhe Diao, Yu Yang, Yonggan Fu, Xin Dong, Dan Su, Markus Kliegl, Zijia Chen, Peter Belcak, Yoshi Suhara, Hongxu Yin, Mostofa Patwary, Yingyan, Lin, Jan Kautz, and Pavlo Molchanov. Climb: Clustering-based iterative data mixture bootstrapping for language model pre-training, 2025. URL https://arxiv.org/abs/2504.13161.

Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

Meng Ding, Kaiyi Ji, Di Wang, and Jinhui Xu. Understanding forgetting in continual learning with linear regression. In *Forty-first International Conference on Machine Learning*, 2024.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051, 2023.

Ding-Zhu Du and Frank Kwang-ming Hwang. *Combinatorial group testing and its applications*, volume 12. World Scientific, 1999.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Pradeep Dubey and Robert J Weber. Probabilistic values for games. Technical report, Cowles Foundation for Research in Economics, 1977.

Pradeep Dubey, Abraham Neyman, and Robert James Weber. Value theory without efficiency. *Mathematics of Operations Research*, 6(1):122–128, 1981.

Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and trends® in theoretical computer science*, 9(3–4):211–407, 2014.

Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, Sep 1936. ISSN 1860-0980. doi:10.1007/BF02288367. URL https://doi.org/10.1007/BF02288367.

Naoki Egami and Kosuke Imai. Causal interaction in factorial experiments: Application to conjoint analysis. *Journal of the American Statistical Association*, 2019.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. https://transformer-circuits.pub/2021/framework/index.html.

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. T-rex: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

Logan Engstrom, Axel Feldmann, and Aleksander Madry. Dsdm: Model-aware dataset selection with datamodels. In *International Conference on Machine Learning*, pages 12491–12526. PMLR, 2024.

Jacob R Epifano, Ravi P Ramachandran, Aaron J Masino, and Ghulam Rasool. Revisiting the fragility of influence functions. *Neural Networks*, 162:581–588, 2023.

Itay Evron, Edward Moroshko, Rachel Ward, Nathan Srebro, and Daniel Soudry. How catastrophic can catastrophic forgetting be in linear regression? In *Conference on Learning Theory*, pages 4028–4079. PMLR, 2022.

Minghong Fang, Neil Zhenqiang Gong, and Jia Liu. Influence function based data poisoning attacks to top-n recommender systems. In *Proceedings of The Web Conference 2020*, pages 3019–3025, 2020.

Abou Zakaria Faroukhi, Imane El Alaoui, Youssef Gahi, and Aouatif Amine. Big data monetization throughout big data value chain: a comprehensive review. *Journal of Big Data*, 7(1):3, 2020.

Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. In *NeurIPS*, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/1e14bfe2714193e7af5abc64ecbd6b46-Abstract.html`.

Ronald A Fisher. On the mathematical foundations of theoretical statistics. *Philosophical transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character*, 222(594-604):309–368, 1922.

Mike Fleckenstein, Ali Obaidi, and Nektaria Tryfona. A review of data valuation approaches and building and scoring a data valuation model. *Harvard Data Science Review*, 5(1), 2023.

Milad Fotouhi, Mohammad Taha Bahadori, Oluwaseyi Feyisetan, Payman Arabshahi, and David Heckerman. Fast training dataset attribution via in-context learning. *arXiv preprint arXiv:2408.11852*, 2024.

Fabian Fumagalli, Maximilian Muschalik, Patrick Kolpaczki, Eyke Hüllermeier, and Barbara Hammer. Shap-iq: Unified approximation of any-order shapley interactions. *Advances in Neural Information Processing Systems*, 36, 2024.

Ruyi Gan, Ziwei Wu, Renliang Sun, Junyu Lu, Xiaojun Wu, Dixiang Zhang, Kunhao Pan, Junqing He, Yuanhe Tian, Ping Yang, Qi Yang, Hao Wang, Jiaxing Zhang, and Yan Song. Ziya2: Data-centric learning is all llms need, 2024. URL `https://arxiv.org/abs/2311.03301`.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Thomas George, César Laurent, Xavier Bouthillier, Nicolas Ballas, and Pascal Vincent. Fast approximate natural gradient descent in a kronecker factored eigenbasis. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL `https://proceedings.neurips.cc/paper_files/paper/2018/file/48000647b315f6f00f913caa757a70b3-Paper.pdf`.

Kristian Georgiev, Joshua Vendrow, Hadi Salman, Sung Min Park, and Aleksander Madry. The journey, not the destination: How data guides diffusion models. *arXiv preprint arXiv:2312.06205*, 2023.

Kristian Georgiev, Roy Rinberg, Sung Min Park, Shivam Garg, Andrew Ilyas, Aleksander Madry, and Seth Neel. Machine unlearning via simulated oracle matching. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=3vXpZpOn29`.

Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. In *International conference on machine learning*, pages 2242–2251. PMLR, 2019.

Amirata Ghorbani, Michael Kim, and James Zou. A distributional framework for data valuation. In *International Conference on Machine Learning*, pages 3535–3544. PMLR, 2020.

Antonio Ginart, Melody Guan, Gregory Valiant, and James Y Zou. Making ai forget you: Data deletion in machine learning. *Advances in neural information processing systems*, 32, 2019.

Ryan Giordano, Michael I Jordan, and Tamara Broderick. A higher-order swiss army infinitesimal jackknife. *arXiv preprint arXiv:1907.12116*, 2019a.

Ryan Giordano, William Stephenson, Runjing Liu, Michael Jordan, and Tamara Broderick. A swiss army infinitesimal jackknife. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1139–1147. PMLR, 2019b.

Aaron Gokaslan and Vanya Cohen. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`, 2019.

Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9304–9312, 2020.

Youdi Gong, Guangzhen Liu, Yunzhi Xue, Rui Li, and Lingzhong Meng. A survey on dataset quality in machine learning. *Information and Software Technology*, 162:107268, 2023.

Ian Goodfellow. *Deep learning*, volume 196. MIT press, 2016.

Sven Gowal, Sylvestre-Alvise Rebuffi, Olivia Wiles, Florian Stimberg, Dan Andrei Calian, and Timothy A Mann. Improving robustness using generated data. *Advances in neural information processing systems*, 34:4218–4233, 2021.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(25):723–773, 2012. URL `http://jmlr.org/papers/v13/gretton12a.html`.

Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, Evan Hubinger, Kamilė Lukošiūtė, Karina Nguyen, Nicholas Joseph, Sam McCandlish, Jared Kaplan, and Samuel R. Bowman. Studying large language model generalization with influence functions, 2023. URL `https://arxiv.org/abs/2308.03296`.

Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified data removal from machine learning models. In *Proceedings of the 37th International Conference on Machine Learning*, pages 3832–3842, 2020.

Han Guo, Nazneen Rajani, Peter Hase, Mohit Bansal, and Caiming Xiong. FastIF: Scalable influence functions for efficient model interpretation and debugging. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10333–10350, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi:10.18653/v1/2021.emnlp-main.808. URL `https://aclanthology.org/2021.emnlp-main.808/`.

Kelvin Guu, Albert Webson, Ellie Pavlick, Lucas Dixon, Ian Tenney, and Tolga Bolukbasi. Simfluence: Modeling the influence of individual training examples by simulating training runs. *arXiv preprint arXiv:2303.08114*, 2023.

Zayd Hammoudeh and Daniel 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*, pages 1367–1381, 2022.

Zayd Hammoudeh and Daniel Lowd. Training data influence analysis and estimation: A survey. *Machine Learning*, 113(5):2351–2403, 2024.

Frank R Hampel. The influence curve and its role in robust estimation. *Journal of the american statistical association*, 69(346):383–393, 1974.

Frank R Hampel, Elvezio M Ronchetti, Peter J Rousseeuw, and Werner A Stahel. *Robust Statistics: The Approach Based on Influence Functions*. John Wiley & Sons, 2011.

Xiaochuang Han and Yulia Tsvetkov. Fortifying toxic speech detectors against veiled toxicity. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7732–7739, Online, November 2020. Association for Computational Linguistics. doi:10.18653/v1/2020.emnlp-main.622. URL `https://aclanthology.org/2020.emnlp-main.622/`.

Xiaochuang Han and Yulia Tsvetkov. Influence tuning: Demoting spurious correlations via instance attribution and instance-driven updates. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4398–4409, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi:10.18653/v1/2021.findings-emnlp.374. URL `https://aclanthology.org/2021.findings-emnlp.374/`.

Xiaochuang Han, Byron C. Wallace, and Yulia Tsvetkov. Explaining black box predictions and unveiling data artifacts through influence functions. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5553–5563, Online, July 2020. Association for Computational Linguistics. doi:10.18653/v1/2020.acl-main.492. URL `https://aclanthology.org/2020.acl-main.492/`.

Xiaochuang Han, Daniel Simig, Todor Mihaylov, Yulia Tsvetkov, Asli Celikyilmaz, and Tianlu Wang. Understanding in-context learning via supportive pretraining data. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume*

1: Long Papers), pages 12660–12673, Toronto, Canada, July 2023. Association for Computational Linguistics. doi:10.18653/v1/2023.acl-long.708. URL https://aclanthology.org/2023.acl-long.708/.

Satoshi Hara, Atsushi Nitanda, and Takanori Maehara. Data cleansing for models trained with sgd. *Advances in Neural Information Processing Systems*, 32, 2019.

Mark Harden. The artchive. https://www.artchive.com/, Accessed 2025.

John C Harsanyi. A simplified bargaining model for the n-person cooperative game. *Papers in game theory*, pages 44–70, 1982.

Luxi He, Mengzhou Xia, and Peter Henderson. What is in your safe data? identifying benign data that breaks safety. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Hi8jKh4HE9.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

J. Herzberger and Lj. Petković. Efficient iterative algorithms for bounding the inverse of a matrix. *Computing*, 44 (3):237–244, September 1990. ISSN 1436-5057. doi:10.1007/BF02262219. URL https://doi.org/10.1007/BF02262219.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020a.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020b.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022a. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Hongsheng Hu, Zoran Salcic, Lichao Sun, Gillian Dobbie, Philip S Yu, and Xuyun Zhang. Membership inference attacks on machine learning: A survey. *ACM Computing Surveys (CSUR)*, 54(11s):1–37, 2022b.

Yuzheng Hu, Pingbang Hu, Han Zhao, and Jiaqi Ma. Most influential subset selection: Challenges, promises, and beyond. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024a. URL https://openreview.net/forum?id=qWi33pPecC.

Yuzheng Hu, Fan Wu, Qinbin Li, Yunhui Long, Gonzalo Munilla Garrido, Chang Ge, Bolin Ding, David Forsyth, Bo Li, and Dawn Song. Sok: Privacy-preserving data synthesis. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 4696–4713. IEEE, 2024b.

Yuzheng Hu, Fan Wu, Haotian Ye, David Forsyth, James Zou, Nan Jiang, Jiaqi W Ma, and Han Zhao. A snapshot of influence: A local data attribution framework for online reinforcement learning. *arXiv preprint arXiv:2505.19281*, 2025.

Kai Hua, Steven Wu, Ge Zhang, and Ke Shen. Attentioninfluence: Adopting attention head influence for weak-to-strong pretraining data selection. *arXiv preprint arXiv:2505.07293*, 2025.

Chengzhi Huang and Hui Li. Single-user injection for invisible shilling attack against recommender systems. In *Proceedings of the 32nd ACM international conference on information and knowledge management*, pages 864–873, 2023.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Trans. Inf. Syst.*, 43(2), January 2025. ISSN 1046-8188. doi:10.1145/3703155. URL https://doi.org/10.1145/3703155.

Zirui Huang, Yunlong Mao, and Sheng Zhong. Uba-inf: unlearning activated backdoor attack with influence-driven camouflage. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 4211–4228, 2024.

Peter J Huber and Elvezio M Ronchetti. *Robust Statistics*. Wiley Series in Probability and Statistics. Wiley-Blackwell, Hoboken, NJ, 2 edition, January 2009. ISBN 9780470129906,9780470434697. doi:10.1002/9780470434697.

Andrew Ilyas, Sung Min Park, Logan Engstrom, Guillaume Leclerc, and Aleksander Madry. Datamodels: Understanding predictions with data and data with predictions. In *International Conference on Machine Learning*, pages 9525–9587. PMLR, 2022.

Guido W Imbens and Donald B Rubin. *Causal inference in statistics, social, and biomedical sciences*. Cambridge university press, 2015.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

Matthew Jagielski, Giorgio Severi, Niklas Pousette Harger, and Alina Oprea. Subpopulation data poisoning attacks. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 3104–3122, 2021.

Himanshu Jahagirdar, Jiachen T Wang, and Ruoxi Jia. Data valuation in the absence of a reliable validation set. *Transactions on Machine Learning Research*, 2024.

Neel Jain, Avi Schwarzschild, Yuxin Wen, Gowthami Somepalli, John Kirchenbauer, Ping yeh Chiang, Micah Goldblum, Aniruddha Saha, Jonas Geiping, and Tom Goldstein. Baseline defenses for adversarial attacks against aligned language models, 2023. URL https://arxiv.org/abs/2309.00614.

Saachi Jain, Kimia Hamidieh, Kristian Georgiev, Andrew Ilyas, Marzyeh Ghassemi, and Aleksander Madry. Improving subgroup robustness via data selection. *Advances in Neural Information Processing Systems*, 37:94490–94511, 2024.

Neil Jethani, Mukund Sudarshan, Ian Connick Covert, Su-In Lee, and Rajesh Ranganath. FastSHAP: Real-time shapley value estimation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=Zq2G_VTV53T.

Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nezihe Merve Gurel, Bo Li4 Ce Zhang, and Costas Spanos1 Dawn Song. Efficient task-specific data valuation for nearest neighbor algorithms. *Proceedings of the VLDB Endowment*, 12(11), 2019a.

Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gürel, Bo Li, Ce Zhang, Dawn Song, and Costas J Spanos. Towards efficient data valuation based on the shapley value. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1167–1176. PMLR, 2019b.

Ruoxi Jia, Fan Wu, Xuehui Sun, Jiacen Xu, David Dao, Bhavya Kailkhura, Ce Zhang, Bo Li, and Dawn 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*, pages 8239–8247, 2021.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

Cathy Jiao, Gary Gao, and Chenyan Xiong. In-context probing approximates influence function for data valuation, 2024. URL https://arxiv.org/abs/2407.12259.

Cathy Jiao, Weizhen Gao, Aditi Raghunathan, and Chenyan Xiong. On the feasibility of in-context probing for data attribution. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 5140–5155, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-7. doi:10.18653/v1/2025.findings-naacl.286. URL https://aclanthology.org/2025.findings-naacl.286/.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8018–8025, 2020.

Yiming Ju, Yuanzhe Zhang, Zhao Yang, Zhongtao Jiang, Kang Liu, and Jun Zhao. Logic traps in evaluating attribution scores. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5911–5922, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.acl-long.407. URL https://aclanthology.org/2022.acl-long.407.

Hoang Anh Just, Feiyang Kang, Tianhao Wang, Yi Zeng, Myeongseob Ko, Ming Jin, and Ruoxi Jia. LAVA: Data valuation without pre-specified learning algorithms. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=JJuP86nBl4q.

Karthikeyan K and Anders Søgaard. Revisiting methods for finding influential examples. *arXiv preprint arXiv:2111.04683*, 2021.

Feiyang Kang, Hoang Anh Just, Anit Kumar Sahu, and Ruoxi Jia. Performance scaling via optimal transport: Enabling data selection from partially revealed sources. *Advances in Neural Information Processing Systems*, 36:61341–61363, 2023.

Feiyang Kang, Yifan Sun, Bingbing Wen, Si Chen, Dawn Song, Rafid Mahmood, and Ruoxi Jia. Autoscale: Automatic prediction of compute-optimal data composition for training llms. *arXiv preprint arXiv:2407.20177*, 2024.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models. In *Forty-first International Conference on Machine Learning*, 2024.

Bojan Karlaš, David Dao, Matteo Interlandi, Sebastian Schelter, Wentao Wu, and Ce Zhang. Data debugging with shapley importance over machine learning pipelines. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=qxGXjWxabq.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Hk99zCeAb.

Phillip Keung, Yichao Lu, György Szarvas, and Noah A Smith. The multilingual amazon reviews corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4563–4568, 2020.

Alaa Khaddaj, Logan Engstrom, and Aleksander Madry. Small-to-large generalization: Training data influences models consistently across scale. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=79ZkWgY2FI.

Rajiv Khanna, Been Kim, Joydeep Ghosh, and Sanmi Koyejo. Interpreting black box predictions using fisher kernels. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3382–3390. PMLR, 2019.

SungYub Kim, Kyungsu Kim, and Eunho Yang. Gex: A flexible method for approximating influence via geometric ensemble. *Advances in Neural Information Processing Systems*, 36:5888–5911, 2023.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Myeongseob Ko, Feiyang Kang, Weiyan Shi, Ming Jin, Zhou Yu, and Ruoxi Jia. The mirrored influence hypothesis: Efficient data influence estimation by harnessing forward passes. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 26276–26285, 2024. doi:10.1109/CVPR52733.2024.02483.

Jungyeon Koh, Hyeonsu Lyu, Jonggyu Jang, and Hyun Jong Yang. Faithful and fast influence function via advanced sampling. In *ICML 2024 Workshop on Mechanistic Interpretability*, 2024. URL https://openreview.net/forum?id=TTVPbaxXjR.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017.

Pang Wei W Koh, Kai-Siang Ang, Hubert Teo, and Percy S Liang. On the accuracy of influence functions for measuring group effects. *Advances in neural information processing systems*, 32, 2019.

Patrick Kolpaczki, Viktor Bengs, Maximilian Muschalik, and Eyke Hüllermeier. Approximating the shapley value without marginal contributions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 13246–13255, 2024.

Shuming Kong, Yanyan Shen, and Linpeng Huang. Resolving training biases via influence-based data relabeling. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=EskfH0bwNVn.

Steven George Krantz and Harold R Parks. *The implicit function theorem: history, theory, and applications*. Springer Science & Business Media, 2002.

Alex Krizhevsky. *Learning Multiple Layers of Features from Tiny Images*. PhD thesis, University of Toronto, Toronto, ON, Canada, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research), 2010. URL http://www.cs.toronto.edu/kriz/cifar.html.

Nupur Kumari, Bingliang Zhang, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. Multi-concept customization of text-to-image diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1931–1941, 2023.

Yongchan Kwon and James Zou. Beta shapley: a unified and noise-reduced data valuation framework for machine learning. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 8780–8802. PMLR, 28–30 Mar 2022. URL https://proceedings.mlr.press/v151/kwon22a.html.

Yongchan Kwon and James Zou. Data-oob: Out-of-bag estimate as a simple and efficient data value. In *International Conference on Machine Learning*, pages 18135–18152. PMLR, 2023.

Yongchan Kwon, Manuel A Rivas, and James Zou. Efficient computation and analysis of distributional shapley values. In *International Conference on Artificial Intelligence and Statistics*, pages 793–801. PMLR, 2021.

Yongchan Kwon, Eric Wu, Kevin Wu, and James Zou. Datainf: Efficiently estimating data influence in loRA-tuned LLMs and diffusion models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=9m02ib92Wz.

Tsz Kin Lam, Eva Hasler, and Felix Hieber. Analyzing the use of influence functions for instance-specific data filtering in neural machine translation. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 295–309, Abu Dhabi, United Arab Emirates (Hybrid), December 2022. Association for Computational Linguistics.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Jinhyuk Lee, Zhuyun Dai, Xiaoqi Ren, Blair Chen, Daniel Cer, Jeremy R. Cole, Kai Hui, Michael Boratko, Rajvi Kapadia, Wen Ding, Yi Luan, Sai Meher Karthik Duddu, Gustavo Hernandez Abrego, Weiqiang Shi, Nithi Gupta, Aditya Kusupati, Prateek Jain, Siddhartha Reddy Jonnalagadda, Ming-Wei Chang, and Iftekhar Naim. Gecko: Versatile text embeddings distilled from large language models, 2024. URL https://arxiv.org/abs/2403.20327.

Dan Ley, Suraj Srinivas, Shichang Zhang, Gili Rusak, and Himabindu Lakkaraju. Generalized group data attribution, 2024. URL https://arxiv.org/abs/2410.09940.

Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. Data poisoning attacks on factorization-based collaborative filtering. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/83fa5a432ae55c253d0e60dbfa716723-Paper.pdf.

Kangming Li, Daniel Persaud, Kamal Choudhary, Brian DeCost, Michael Greenwood, and Jason Hattrick-Simpers. Exploiting redundancy in large materials datasets for efficient machine learning with less data. *Nature Communications*, 14(1):7283, 2023.

Tianjian Li, Haoran Xu, Philipp Koehn, Daniel Khashabi, and Kenton Murray. Error norm truncation: Robust training in the presence of data noise for text generation models. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=zMvMwNvs4R.

Weida Li and Yaoliang Yu. Robust data valuation with weighted banzhaf values. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 60349–60383. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/bdb0596d13cfccf2db6f0cc5280d2a3f-Paper-Conference.pdf.

Weida Li and Yaoliang Yu. Faster approximation of probabilistic and distributional values via least squares. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=lvSMIsztka.

Weida Li and Yaoliang Yu. One sample fits all: Approximating all probabilistic values simultaneously and efficiently. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 58309–58340. Curran Associates, Inc., 2024b. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/6b295b08549c0441914e391651423477-Paper-Conference.pdf.

Xuechen Li, Florian Tramer, Percy Liang, and Tatsunori Hashimoto. Large language models can be strong differentially private learners. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=bVuP3ltATMz.

Yanwei Li, Yuechen Zhang, Chengyao Wang, Zhisheng Zhong, Yixin Chen, Ruihang Chu, Shaoteng Liu, and Jiaya Jia. Mini-gemini: Mining the potential of multi-modality vision language models. *arXiv preprint arXiv:2403.18814*, 2024b.

Yunshui Li, Binyuan Hui, Xiaobo Xia, Jiaxi Yang, Min Yang, Lei Zhang, Shuzheng Si, Ling-Hao Chen, Junhao Liu, Tongliang Liu, Fei Huang, and Yongbin Li. One-shot learning as instruction data prospector for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4586–4601, 2024c.

Fan Liang, Wei Yu, Dou An, Qingyu Yang, Xinwen Fu, and Wei Zhao. A survey on big data market: Pricing, trading and protection. *IEEE Access*, 6:15132–15154, 2018. doi:10.1109/ACCESS.2018.2806881.

Peiyuan Liao, Xiuyu Li, Xihui Liu, and Kurt Keutzer. The artbench dataset: Benchmarking generative models with artworks. *arXiv preprint arXiv:2206.11404*, 2022.

Chris Lin, Mingyu Lu, and Su-In Lee. Efficient global data attribution for diffusion models. In *Workshop on Navigating and Addressing Data Problems for Foundation Models at ICLR*, 2024a.

Huawei Lin, Jikai Long, Zhaozhuo Xu, and Weijie Zhao. Token-wise influential training data retrieval for large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 841–860, Bangkok, Thailand, August 2024b. Association for Computational Linguistics. doi:10.18653/v1/2024.acl-long.48. URL https://aclanthology.org/2024.acl-long.48/.

Jinkun Lin, Anqi Zhang, Mathias Lécuyer, Jinyang Li, Aurojit Panda, and Siddhartha Sen. Measuring the effect of training data on deep learning predictions via randomized experiments. In *International Conference on Machine Learning*, pages 13468–13504. PMLR, 2022.

Jinxu Lin, Linwei Tao, Minjing Dong, and Chang Xu. Diffusion attribution score: Evaluating training data influence in diffusion model. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=kuutidLf6R.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

Xiaoqiang Lin, Xinyi Xu, Zhaoxuan Wu, See-Kiong Ng, and Bryan Kian Hsiang Low. Distributionally robust data valuation. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 30362–30391. PMLR, 21–27 Jul 2024c. URL https://proceedings.mlr.press/v235/lin24t.html.

Zi Lin, Zihan Wang, Yongqi Tong, Yangkun Wang, Yuxin Guo, Yujia Wang, and Jingbo Shang. Toxicchat: Unveiling hidden challenges of toxicity detection in real-world user-ai conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4694–4702, 2023.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 34892–34916. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/6dcf277ea32ce3288914faf369fe6de0-Paper-Conference.pdf.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 26296–26306, 2024a.

Jinxin Liu and Zao Yang. Tracing privacy leakage of language models to training data via adjusted influence functions. *arXiv preprint arXiv:2408.10468*, 2024.

Qian Liu, Xiaosen Zheng, Niklas Muennighoff, Guangtao Zeng, Longxu Dou, Tianyu Pang, Jing Jiang, and Min Lin. Regmix: Data mixture as regression for language model pre-training. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=5BjQOUXq7i.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Zelei Liu, Yuanyuan Chen, Han Yu, Yang Liu, and Lizhen Cui. Gtg-shapley: Efficient and accurate participant contribution evaluation in federated learning. *ACM Transactions on intelligent Systems and Technology (TIST)*, 13 (4):1–21, 2022.

Zikang Liu, Kun Zhou, Wayne Xin Zhao, Dawei Gao, Yaliang Li, and Ji-Rong Wen. Less is more: High-value data selection for visual instruction tuning, 2024b. URL https://arxiv.org/abs/2403.09559.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Elita Lobo, Harvineet Singh, Marek Petrik, Cynthia Rudin, and Himabindu Lakkaraju. Data poisoning attacks on off-policy policy evaluation methods. In James Cussens and Kun Zhang, editors, *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pages 1264–1274. PMLR, 01–05 Aug 2022. URL https://proceedings.mlr.press/v180/lobo22a.html.

Shayne Longpre, Gregory Yauney, Emily Reif, Katherine Lee, Adam Roberts, Barret Zoph, Denny Zhou, Jason Wei, Kevin Robinson, David Mimno, and Daphne Ippolito. A pretrainer's guide to training data: Measuring the effects

of data age, domain coverage, quality, & toxicity. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3245–3276, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi:10.18653/v1/2024.naacl-long.179. URL https://aclanthology.org/2024.naacl-long.179/.

MingYu Lu, Chris Lin, Chanwoo Kim, and Su-In Lee. An efficient framework for crediting data contributors of diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=9EqQC2ct4H.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, 30, 2017.

Sadhika Malladi, Alexander Wettig, Dingli Yu, Danqi Chen, and Sanjeev Arora. A kernel-based view of language model fine-tuning. In *International Conference on Machine Learning*, pages 23610–23641. PMLR, 2023.

Alessandro Mantelero. The eu proposal for a general data protection regulation and the roots of the 'right to be forgotten'. *Computer Law & Security Review*, 29(3):229–235, 2013. ISSN 0267-3649. doi:https://doi.org/10.1016/j.clsr.2013.03.010.

Neil G Marchant, Benjamin IP Rubinstein, and Scott Alfeld. Hard to forget: Poisoning attacks on certified machine unlearning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 7691–7700, 2022.

James Martens. New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research*, 21(146):1–76, 2020.

James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2408–2417, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/martens15.html.

Jacob Menick, Maja Trebacz, Vladimir Mikulik, John Aslanides, Francis Song, Martin Chadwick, Mia Glaese, Susannah Young, Lucy Campbell-Gillingham, Geoffrey Irving, et al. Teaching language models to support answers with verified quotes. *arXiv preprint arXiv:2203.11147*, 2022.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Byj72udxe.

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *International Conference on Learning Representations*, 2013. URL https://api.semanticscholar.org/CorpusID:5959482.

Bruno Kacper Mlodozeniec, Runa Eschenhagen, Juhan Bae, Alexander Immer, David Krueger, and Richard E. Turner. Influence functions for scalable data attribution in diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=esYrEndGsr.

Nicholas Moehlea, Stephen Boydb, and Andrew Angc. Portfolio performance attribution via shapley value. *Journal Of Investment Management*, 20(3):33–52, 2022.

Ankur Moitra and Dhruv Rohatgi. Provably auditing ordinary least squares in low dimensions. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=DlpCotqdTy.

John X. Morris, Chawin Sitawarin, Chuan Guo, Narine Kokhlikyan, G. Edward Suh, Alexander M. Rush, Kamalika Chaudhuri, and Saeed Mahloujifar. How much do language models memorize?, 2025a. URL https://arxiv.org/abs/2505.24832.

John X Morris, Chawin Sitawarin, Chuan Guo, Narine Kokhlikyan, G Edward Suh, Alexander M Rush, Kamalika Chaudhuri, and Saeed Mahloujifar. How much do language models memorize? *arXiv preprint arXiv:2505.24832*, 2025b.

Mohammad S Najjar and William J Kettinger. Data monetization: Lessons from a retailer's journey. *MIS Quarterly Executive*, 12(4), 2013.

Seth Neel, Aaron Roth, and Saeed Sharifi-Malvajerdi. Descent-to-delete: Gradient-based methods for machine unlearning. In *Algorithmic Learning Theory*, pages 931–962. PMLR, 2021.

Elisa Nguyen, Minjoon Seo, and Seong Joon Oh. A bayesian approach to analysing training data attribution in deep learning. *Advances in Neural Information Processing Systems*, 36:64155–64180, 2023.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.

Konstantin D Pandl, Fabian Feiland, Scott Thiebes, and Ali Sunyaev. Trustworthy machine learning for health care: scalable data valuation with the shapley value. In *Proceedings of the Conference on Health, Inference, and Learning*, pages 47–57, 2021.

Mathias PM Parisot, Balazs Pejo, and Dayana Spagnuelo. Property inference attacks on convolutional neural networks: Influence and implications of target model's complexity. *arXiv preprint arXiv:2104.13061*, 2021.

Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. Trak: Attributing model behavior at scale. In *International Conference on Machine Learning*, pages 27074–27113. PMLR, 2023.

Barak A Pearlmutter. Fast exact multiplication by the hessian. *Neural computation*, 6(1):147–160, 1994.

Karl Pearson. Vii. note on regression and inheritance in the case of two parents. *proceedings of the royal society of London*, 58(347-352):240–242, 1895.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Pouya Pezeshkpour, Sarthak Jain, Sameer Singh, and Byron Wallace. Combining feature and instance attribution to detect artifacts. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1934–1946, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.findings-acl.153. URL https://aclanthology.org/2022.findings-acl.153/.

Lutz Prechelt. Early stopping-but when? In *Neural Networks: Tricks of the trade*, pages 55–69. Springer, 2002.

Daryl Pregibon. Logistic regression diagnostics. *The annals of statistics*, 9(4):705–724, 1981.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. *Advances in Neural Information Processing Systems*, 33:19920–19930, 2020.

Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. Fine-tuning aligned language models compromises safety, even when users do not intend to! In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=hTEGyKf0dZ.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PmLR, 2021.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Nazneen Fatema Rajani, Ben Krause, Wengpeng Yin, Tong Niu, Richard Socher, and Caiming Xiong. Explaining and improving model behavior with k nearest neighbor representations. *arXiv preprint arXiv:2010.09030*, 2020.

Juan Ramos et al. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 29–48. Citeseer, 2003.

Ramesh Raskar, Praneeth Vepakomma, Tristan Swedish, and Aalekh Sharan. Data markets to support ai for all: Pricing, valuation and governance. *arXiv preprint arXiv:1905.06462*, 2019.

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. Okapi at trec-3. In *Text Retrieval Conference*, 1994. URL https://api.semanticscholar.org/CorpusID:41563977.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

Paul Röttger, Hannah Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. Xstest: A test suite for identifying exaggerated safety behaviours in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5377–5400, 2024.

Benedek Rozemberczki, Lauren Watson, Péter Bayer, Hao-Tsung Yang, Olivér Kiss, Sebastian Nilsson, and Rik Sarkar. The shapley value in machine learning. *arXiv preprint arXiv:2202.05594*, 2022.

Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. In *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019*. The Internet Society, 2019.

Ayrton San Joaquin, Bin Wang, Zhengyuan Liu, Nicholas Asher, Brian Lim, Philippe Muller, and Nancy F. Chen. In2Core: Leveraging influence functions for coreset selection in instruction finetuning of large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10324–10335, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi:10.18653/v1/2024.findings-emnlp.604. URL https://aclanthology.org/2024.findings-emnlp.604/.

Andrea Schioppa, Polina Zablotskaia, David Vilar, and Artem Sokolov. Scaling up influence functions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8179–8186, 2022.

Andrea Schioppa, Katja Filippova, Ivan Titov, and Polina Zablotskaia. Theoretical and practical perspectives on what influence functions do. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=gGl0n7Onug.

Stephanie Schoch, Haifeng Xu, and Yangfeng Ji. CS-shapley: Class-wise shapley values for data valuation in classification. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=KTOcrOR5mQ9.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in neural information processing systems*, 35:25278–25294, 2022.

Andrea Sestino, Adham Kahlawi, and Andrea De Mauro. Decoding the data economy: a literature review of its impact on business, society and digital transformation. *European Journal of Innovation Management*, 28(2):298–323, 2025.

Lloyd S Shapley. A value for n-person games. *Contribution to the Theory of Games*, 2, 1953.

Lee Sharkey, Bilal Chughtai, Joshua Batson, Jack Lindsey, Jeff Wu, Lucius Bushnaq, Nicholas Goldowsky-Dill, Stefan Heimersheim, Alejandro Ortega, Joseph Bloom, Stella Biderman, Adria Garriga-Alonso, Arthur Conmy, Neel Nanda, Jessica Rumbelow, Martin Wattenberg, Nandi Schoots, Joseph Miller, Eric J. Michaud, Stephen Casper, Max Tegmark, William Saunders, David Bau, Eric Todd, Atticus Geiger, Mor Geva, Jesse Hoogland, Daniel Murfet, and Tom McGrath. Open problems in mechanistic interpretability, 2025. URL https://arxiv.org/abs/2501.16496.

Dongsub Shim, Zheda Mai, Jihwan Jeong, Scott Sanner, Hyunwoo Kim, and Jongseong Jang. Online class-incremental continual learning with adversarial shapley value. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9630–9638, 2021.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

Rachael Hwee Ling Sim, Xinyi Xu, and Bryan Kian Hsiang Low. Data valuation in machine learning:" ingredients", strategies, and open challenges. In *IJCAI*, pages 5607–5614, 2022.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=St1giarCHLP.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. *Advances in neural information processing systems*, 34:1415–1428, 2021b.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021c.

Charles Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72–101, 1904.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Mirac Suzgun, Nathan Scales, Nathanael Scharli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, 2023.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html*, 3(6):7, 2023.

Sebastian Shenghong Tay, Xinyi Xu, Chuan Sheng Foo, and Bryan Kian Hsiang Low. Incentivizing collaboration in machine learning via synthetic data rewards. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 9448–9456, 2022.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Sam Bowman, Dipanjan Das, and Ellie Pavlick. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=SJzSgnRcKX`.

Naoyuki Terashita, Hiroki Ohashi, Yuichi Nonaka, and Takashi Kanemaru. Influence estimation for generative adversarial networks. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=opHLcXxYTC_`.

Stefano Teso, Andrea Bontempelli, Fausto Giunchiglia, and Andrea Passerini. Interactive label cleaning with example-based explanations. *Advances in Neural Information Processing Systems*, 34:12966–12977, 2021.

Megh Thakkar, Tolga Bolukbasi, Sriram Ganapathy, Shikhar Vashishth, Sarath Chandar, and Partha Talukdar. Self-influence guided data reweighting for language model pre-training. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2033–2045, Singapore, December 2023. Association for Computational Linguistics.

Zhihua Tian, Jian Liu, Jingyu Li, Xinle Cao, Ruoxi Jia, Jun Kong, Mengdi Liu, and Kui Ren. Private data valuation and fair payment in data marketplaces. *arXiv preprint arXiv:2210.08723*, 2022.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996.

Jörg Tiedemann. Parallel data, tools and interfaces in opus. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, 2012.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023b.

Stacey Truex, Ling Liu, Mehmet Emre Gursoy, Lei Yu, and Wenqi Wei. Demystifying membership inference attacks in machine learning as a service. *IEEE transactions on services computing*, 14(6):2073–2089, 2019.

Binghui Wang, Minhua Lin, Tianxiang Zhou, Pan Zhou, Ang Li, Meng Pang, Hai Li, and Yiran Chen. Efficient, direct, and restricted black-box graph evasion attacks to any-layer graph neural networks via influence function. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 693–701, 2024a.

Haonan Wang, Ziwei Wu, and Jingrui He. Fairif: Boosting fairness in deep learning via influence functions with validation set sensitive attributes. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 721–730, 2024b.

Jiachen T Wang and Ruoxi Jia. Data banzhaf: A robust data valuation framework for machine learning. In *International Conference on Artificial Intelligence and Statistics*, pages 6388–6421. PMLR, 2023a.

Jiachen T Wang and Ruoxi Jia. A note on "towards efficient data valuation based on the shapley value". *arXiv preprint arXiv:2302.11431*, 2023b.

Jiachen T Wang, Zhun Deng, Hiroaki Chiba-Okabe, Boaz Barak, and Weijie J Su. An economic solution to copyright challenges of generative ai. *arXiv preprint arXiv:2404.13964*, 2024c.

Jiachen T Wang, Prateek Mittal, and Ruoxi Jia. Efficient data shapley for weighted nearest neighbor algorithms. In *International Conference on Artificial Intelligence and Statistics*, pages 2557–2565. PMLR, 2024d.

Jiachen T. Wang, Tong Wu, Dawn Song, Prateek Mittal, and Ruoxi Jia. GREATS: Online selection of high-quality data for LLM training in every iteration. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024e.

Jiachen T. Wang, Tianji Yang, James Zou, Yongchan Kwon, and Ruoxi Jia. Rethinking data shapley for data selection tasks: misleads and merits. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024f.

Jiachen T. Wang, Prateek Mittal, Dawn Song, and Ruoxi Jia. Data shapley in one training run. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=HD6bWcj87Y.

Jiachen T. Wang, Dawn Song, James Zou, Prateek Mittal, and Ruoxi Jia. Capturing the temporal dependence of training data influence. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=uHLgDEgiS5.

Jingtan Wang, Xinyang Lu, Zitong Zhao, Zhongxiang Dai, Chuan-Sheng Foo, See-Kiong Ng, and Bryan Kian Hsiang Low. Source attribution for large language model-generated data. *arXiv preprint arXiv:2310.00646*, 2023a.

Jingtan Wang, Xiaoqiang Lin, Rui Qiao, Chuan-Sheng Foo, and Bryan Kian Hsiang Low. Helpful or harmful data? fine-tuning-free shapley attribution for explaining language model predictions. In *Forty-first International Conference on Machine Learning*, 2024g. URL https://openreview.net/forum?id=WSpPC1Jm0p.

Sheng-Yu Wang, Alexei A Efros, Jun-Yan Zhu, and Richard Zhang. Evaluating data attribution for text-to-image models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7192–7203, 2023b.

Sheng-Yu Wang, Aaron Hertzmann, Alexei A. Efros, Jun-Yan Zhu, and Richard Zhang. Data attribution for text-to-image models by unlearning synthesized images. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 4235–4266. Curran Associates, Inc., 2024h. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/07fbde96bee50f4e09303fd4f877c2f3-Paper-Conference.pdf.

Xiao Wang, Weikang Zhou, Qi Zhang, Jie Zhou, SongYang Gao, Junzhe Wang, Menghan Zhang, Xiang Gao, Yun Wen Chen, and Tao Gui. Farewell to aimless large-scale pretraining: Influential subset selection for language model. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 555–568, Toronto, Canada, 2023c. Association for Computational Linguistics. URL https://aclanthology.org/2023.findings-acl.35/.

Xinhe Wang, Pingbang Hu, Junwei Deng, and Jiaqi W Ma. Adversarial attacks on data attribution. In *The Thirteenth International Conference on Learning Representations*, 2025c.

Xinyi Wang, Antonis Antoniades, Yanai Elazar, Alfonso Amayuelas, Alon Albalak, Kexun Zhang, and William Yang Wang. Generalization v.s. memorization: Tracing language models' capabilities back to pretraining data. In *The Thirteenth International Conference on Learning Representations*, 2025d. URL https://openreview.net/forum?id=IQxBDLmVpT.

B Johnson William and Joram Lindenstrauss. Extensions of lipschitz mapping into hilbert space. *Contemporary mathematics*, 26(189-206):323, 1984.

Mike Wojnowicz, Ben Cruz, Xuan Zhao, Brian Wallace, Matt Wolff, Jay Luan, and Caleb Crable. "influence sketching": Finding influential samples in large-scale regressions. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 3601–3612. IEEE, 2016.

Chenwang Wu, Defu Lian, Yong Ge, Zhihao Zhu, and Enhong Chen. Triple adversarial learning for influence based poisoning attack in recommender systems. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1830–1840, 2021.

Chenwang Wu, Defu Lian, Yong Ge, Zhihao Zhu, and Enhong Chen. Influence-driven data poisoning for robust recommender systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(10):11915–11931, 2023a.

Jingfeng Wu, Difan Zou, Vladimir Braverman, Quanquan Gu, and Sham Kakade. The power and limitation of pretraining-finetuning for linear regression under covariate shift. *Advances in Neural Information Processing Systems*, 35:33041–33053, 2022a.

Jingfeng Wu, Difan Zou, Zixiang Chen, Vladimir Braverman, Quanquan Gu, and Peter Bartlett. How many pretraining tasks are needed for in-context learning of linear regression? In *The Twelfth International Conference on Learning Representations*, 2024a.

Kangxi Wu, Liang Pang, Huawei Shen, and Xueqi Cheng. Enhancing training data attribution for large language models with fitting error consideration. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 14131–14143, Miami, Florida, USA, November 2024b. Association for Computational Linguistics. doi:10.18653/v1/2024.emnlp-main.782. URL https://aclanthology.org/2024.emnlp-main.782/.

Mengmeng Wu, Ruoxi Jia, Changle Lin, Wei Huang, and Xiangyu Chang. Variance reduced shapley value estimation for trustworthy data valuation. *Computers & Operations Research*, 159:106305, 2023b.

Zhaoxuan Wu, Yao Shu, and Bryan Kian Hsiang Low. Davinz: Data valuation using deep neural networks at initialization. In *International Conference on Machine Learning*, pages 24150–24176. PMLR, 2022b.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: Selecting influential data for targeted instruction tuning. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=PG5fV50maR.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy S Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. Doremi: Optimizing data mixtures speeds up language model pretraining. *Advances in Neural Information Processing Systems*, 36:69798–69818, 2023.

Tong Xie, Haoyu Li, Andrew Bai, and Cho-Jui Hsieh. Data attribution for diffusion models: Timestep-induced bias in influence estimation. *Transactions on Machine Learning Research*, 2024a.

Yueqi Xie, Minghong Fang, Renjie Pi, and Neil Gong. GradSafe: Detecting jailbreak prompts for LLMs via safety-critical gradient analysis. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 507–518, Bangkok, Thailand, August 2024b. Association for Computational Linguistics. doi:10.18653/v1/2024.acl-long.30. URL https://aclanthology.org/2024.acl-long.30/.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mT5: A massively multilingual pre-trained text-to-text transformer. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online, June 2021. Association for Computational Linguistics. doi:10.18653/v1/2021.naacl-main.41. URL https://aclanthology.org/2021.naacl-main.41/.

Jiaxi Yang, Wenlong Deng, Benlin Liu, Yangsibo Huang, James Zou, and Xiaoxiao Li. GMValuator: Similarity-based data valuation for generative models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=WncnpvJk83.

Jinghan Yang, Sarthak Jain, and Byron C. Wallace. How many and which training points would need to be removed to flip this prediction? In Andreas Vlachos and Isabelle Augenstein, editors, *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2571–2584, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics. doi:10.18653/v1/2023.eacl-main.188. URL https://aclanthology.org/2023.eacl-main.188/.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. Paws-x: A cross-lingual adversarial dataset for paraphrase identification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3687–3692, 2019.

Yu Yang, Siddhartha Mishra, Jeffrey N Chiang, and Baharan Mirzasoleiman. Smalltolarge (s2l): Scalable data selection for fine-tuning large language models by summarizing training trajectories of small models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=K9IGlMQpif.

Jingwen Ye, Ruonan Yu, Songhua Liu, and Xinchao Wang. Distilled datamodel with reverse gradient matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11954–11963, 2024.

Chih-Kuan Yeh, Joon Kim, Ian En-Hsu Yen, and Pradeep K Ravikumar. Representer point selection for explaining deep neural networks. *Advances in neural information processing systems*, 31, 2018.

Chih-Kuan Yeh, Ankur Taly, Mukund Sundararajan, Frederick Liu, and Pradeep Ravikumar. First is better than last for language data influence. *Advances in Neural Information Processing Systems*, 35:32285–32298, 2022.

Sibo Yi, Yule Liu, Zhen Sun, Tianshuo Cong, Xinlei He, Jiaxing Song, Ke Xu, and Qi Li. Jailbreak attacks and defenses against large language models: A survey. *CoRR*, abs/2407.04295, 2024. URL https://doi.org/10.48550/arXiv.2407.04295.

Jinsung Yoon, Sercan Arik, and Tomas Pfister. Data valuation using reinforcement learning. In *International Conference on Machine Learning*, pages 10842–10851. PMLR, 2020.

Tjalling J Ypma. Historical development of the newton–raphson method. *SIAM review*, 37(4):531–551, 1995.

Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, Sergey Yekhanin, and Huishuai Zhang. Differentially private fine-tuning of language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=Q42f0dfjECO.

Zichun Yu, Spandan Das, and Chenyan Xiong. MATES: Model-aware data selection for efficient pretraining with data influence models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=6gzPSMUAz2.

Chi Zhang, Huaping Zhong, Kuan Zhang, Chengliang Chai, Rui Wang, Xinlin Zhuang, Tianyi Bai, Qiu Jiantao, Lei Cao, Ju Fan, Ye Yuan, Guoren Wang, and Conghui He. Harnessing diversity for important data selection in pretraining large language models. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=bMC1t7eLRc.

Jiayao Zhang, Qiheng Sun, Jinfei Liu, Li Xiong, Jian Pei, and Kui Ren. Efficient sampling approaches to shapley value approximation. *Proceedings of the ACM on Management of Data*, 1(1):1–24, 2023a.

Jiayao Zhang, Yuran Bi, Mengye Cheng, Jinfei Liu, Kui Ren, Qiheng Sun, Yihang Wu, Yang Cao, Raul Castro Fernandez, Haifeng Xu, Ruoxi Jia, Yongchan Kwon, Jian Pei, Jiachen T. Wang, Haocheng Xia, Li Xiong, Xiaohui Yu, and James Zou. A survey on data markets. *CoRR*, abs/2411.07267, 2024. URL https://doi.org/10.48550/arXiv.2411.07267.

Luyang Zhang, Cathy Jiao, Beibei Li, and Chenyan Xiong. Fairshare data pricing for large language models. *arXiv preprint arXiv:2502.00198*, 2025b.

Mengxiao Zhang, Fernando Beltrán, and Jiamou Liu. A survey of data pricing for data marketplaces. *IEEE Transactions on Big Data*, 9(4):1038–1056, 2023b.

Rui Zhang and Shihua Zhang. Rethinking influence functions of neural networks in the over-parameterized regime. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 9082–9090, 2022.

Shichang Zhang, Tessa Han, Usha Bhalla, and Himabindu Lakkaraju. Building bridges, not walls: Advancing interpretability by unifying feature, data, and model component attribution. In *ICLR 2025 Workshop on Navigating and Addressing Data Problems for Foundation Models*, 2025c.

Yuqi Zhang, Liang Ding, Lefei Zhang, and Dacheng Tao. Intention analysis makes LLMs a good jailbreak defender. In Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio, and Steven Schockaert, editors, *Proceedings of the 31st International Conference on Computational Linguistics*, pages 2947–2968, Abu Dhabi, UAE, January 2025d. Association for Computational Linguistics. URL https://aclanthology.org/2025.coling-main.199/.

Zhuo Zhang, Guangyu Shen, Guanhong Tao, Siyuan Cheng, and Xiangyu Zhang. Make them spill the beans! coercive knowledge extraction from (production) llms. *CoRR*, abs/2312.04782, 2023c. URL https://doi.org/10.48550/arXiv.2312.04782.

Bo Zhao, Boya Wu, Muyang He, and Tiejun Huang. Svit: Scaling up visual instruction tuning. *arXiv preprint arXiv:2307.04087*, 2023.

Wanru Zhao, Yaxin Du, Nicholas Donald Lane, Siheng Chen, and Yanfeng Wang. Enhancing data quality in federated fine-tuning of foundation models. In *Workshop on Navigating and Addressing Data Problems for Foundation Models at ICLR*, 2024.

Xiaosen Zheng, Tianyu Pang, Chao Du, Jing Jiang, and Min Lin. Intriguing properties of data attribution on diffusion models. *arXiv preprint arXiv:2311.00500*, 2023.

Xinyu Zhou, Simin Fan, and Martin Jaggi. Hyperinf: Unleashing the hyperpower of the schulz's method for data influence estimation, 2024. URL https://arxiv.org/abs/2410.05090.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models, 2023. URL https://arxiv.org/abs/2307.15043.

Difan Zou, Jingfeng Wu, Vladimir Braverman, Quanquan Gu, and Sham Kakade. Benign overfitting of constant-stepsize sgd for linear regression. In *Conference on Learning Theory*, pages 4633–4635. PMLR, 2021.

A Notation

Table 20: Common notation used throughout the paper.

Symbol	Meaning
$z_i = (x_i, y_i)$	i-th training example (input, label)
$z_{\text{test}}^{(j)}$	(j-th) Query/test example
$D = \{z_i\}$	Training Dataset
$D_{\text{eval}} = \{z_{\text{test}}^{(j)}\}$	Query/test dataset
$S \subseteq D$	Data subset
$f(S; \theta)$	Target function (usually depends on some data S and model θ)
$\hat{\theta}$	Model parameters after training
$\ell(z; \theta)$	Per-example loss function
$\mathcal{L}(D; \theta) = \frac{1}{n} \sum_{z_i \in D} \ell(z_i; \theta)$	Empirical risk (training objective)
$g_i = \nabla_\theta \ell(z_i, \theta)$	Gradient of loss for z_i
H_θ	Hessian of $L(\theta)$
$H_\theta^{(\lambda)} = H_\theta + \lambda I$	Damped Hessian (regularized)
F	Fisher Information Matrix
F_l	Block-diagonal FIM for layer l
G	Stacked gradient matrix $[g_1, \ldots, g_n]^\top$
P	Projection matrix (e.g., random projection)
$\phi_i = P^\top g_i$	Projected gradient for z_i
$\Phi = [\phi_1, \ldots, \phi_n]^\top$	Stacked projected gradients
$\tau_{\texttt{TDA}}(z_i, z_{\text{test}})$	Influence of z_i on query z_{test} under TDA method `TDA`
L	Number of layers in the model
p	Number of model parameters
d	Parameters per layer
r	Reduced dimension after projection
n	Number of training examples
m	Number of query/test examples
B	Batch size
t	Training step or checkpoint index

B Derivations

B.1 Derivation of Banzhaf value

Without loss of generality, define $M = X^\top W X \in \mathbb{R}^{n \times n}$ and $X_j := 2\mathbf{1}_S(i) - 1$. We transform X into a matrix with elements of 1 or -1. Then we have

$$M_{i,j} = \frac{1}{2^n} \sum_{S \subseteq [n]} (2\mathbf{1}_S(i) - 1)(2\mathbf{1}_S(j) - 1) = \begin{cases} 2^{-1}, & \text{if } i = j, \\ 0, & \text{if } i \neq j. \end{cases}$$

Therefore, it's easy to see

$$M^{-1} = 2I$$

and

$$\theta_j^* = 2c_j, \tag{20}$$

where $c_j := \frac{1}{2^n} \sum_S (2\mathbf{1}_{\{j \in S\}} - 1) f_A(z; S)$. We can rewrite the above solution as

$$\theta_j^* = \frac{1}{2^{n-1}} \Big(\sum_{S; j \notin S} f_A(z; S \cup \{j\}) - \sum_{S; j \notin S} f_A(z; S) \Big). \tag{21}$$

In this case, θ_j^* is the Banzhaf value.