Z0-Inf: Zeroth Order Approximation for Data Influence

Narine Kokhlikyan
FAIR at Meta
narine@meta.com

Diego Garcia-Olano
Meta AI
diegoolano@meta.com

Kamalika Chaudhuri *
Google DeepMind
kamalikac@google.com

Saeed Mahloujifar*
Google DeepMind
smahloujifar@google.com

Abstract

Influence functions are a popular tool for measuring the impact of a training point on a model's prediction; in particular, self-influence, which quantifies the influence of a training point on itself, has found many uses such as data selection and outlier detection. However, the use of influence functions has been severely limited in modern models such as LLMs due to their low accuracy or high computational cost: most existing algorithms are either highly inaccurate, or require computing gradients or approximations to inverse Hessians, which can be prohibitive for large models. In this work, we introduce a highly efficient zeroth-order approximation to data influence that requires *only a fraction* of the time and memory footprint of previous methods and is applicable to both differentiable and non-differentiable loss functions. We demonstrate that in addition to its computational advantage, our method delivers superior accuracy in estimating self-influence and comparable or better accuracy in estimating train-test influence for data selection tasks, paving the way for broader and more practical application of influence-function techniques in state-of-the-art AI systems.

The code is available at `https://github.com/facebookresearch/z0inf`.

1 Introduction

Influence functions [Hampel et al., 1986] are a concept from robust statistics that helps us understand how small changes in the data can impact a statistical estimator; in recent years, they have been heavily used in machine learning to measure the impact of particular training data points on model predictions [Koh and Liang, 2020]. One use case is train-test influence that characterizes the degree to which a training data point affects the prediction on a test sample, providing a principled tool for model debugging and dataset refinement. A second important use-case is self-influence, which estimates the degree to which a training point contributes to the model's prediction on itself. This is used in applications, such as, selecting high quality [Xia et al., 2024] or mislabeled data, estimating plausible memorization [Feldman and Zhang, 2020], and detecting outliers [Hara et al., 2019]

Ideally, the estimation of data influence involves finding how much a model's prediction on a data point x_t depends on including a training point x_s – in other words, how much the removal of x_s impacts the model's prediction. This is a particularly computationally challenging problem: calculating the ideal influence on a single data point means re-training a full model! Hence, prior work has looked at finding better methods that are both computationally efficient [Choe et al., 2024, Kwon et al., 2023] and yet provide a good approximation to this ideal "remove-and-retrain" baseline.

*Work done at Meta.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

To this effect, prior work can be divided into two classes of methods. The first, that we call second-order methods, is loosely motivated by the fact that perturbing a loss function by adding a small amount $\epsilon\ell(\theta, x_i)$ to it changes its minimizer θ^* by approximately the product of the inverse Hessian of the loss times the gradient at θ^* [Koh and Liang, 2020]. The goal is then to approximate this product using computationally efficient algorithms [Grosse et al., 2023, Park et al., 2023, Schioppa et al., 2021, Choe et al., 2024]. While algorithmic innovations have addressed the efficiency challenges, a major challenge with this class of methods is that they can diverge considerably from the "remove-and-retrain" baseline, as shown by several authors [Basu et al., 2021, Li et al., 2024, K and Søgaard, 2021, Schioppa et al., 2023]. The second class of methods, exemplified by TracIn [Pruthi et al., 2020] and its variants, leverages the training dynamics through multiple model checkpoints, and uses the gradient as a first order approximation to measure the loss difference across these checkpoints. For larger models, this leads to loss in both accuracy and running time: first-order approximations by gradients can be relatively inaccurate, leading to loss in accuracy, and gradients are compute-intensive, particularly for large models.

In this work, we propose a new method, Z0-Inf, that addresses both challenges. The key insight behind it is the observation that the gradient is a poor first-order approximation to the difference in losses between two checkpoints, especially for larger and more complex models. Instead, we propose to by-pass the gradient altogether, use the finite difference directional gradient and aggregate it across multiple epochs, similar to the methodology in TracIn, resulting in a significantly more accurate algorithm. Additionally, the finite difference directional gradient can only be computed from zeroth-order information, making it considerably faster for large models. Theoretically, we show that our method forms a zeroth-order approximation to TracIn. Empirically, we evaluate

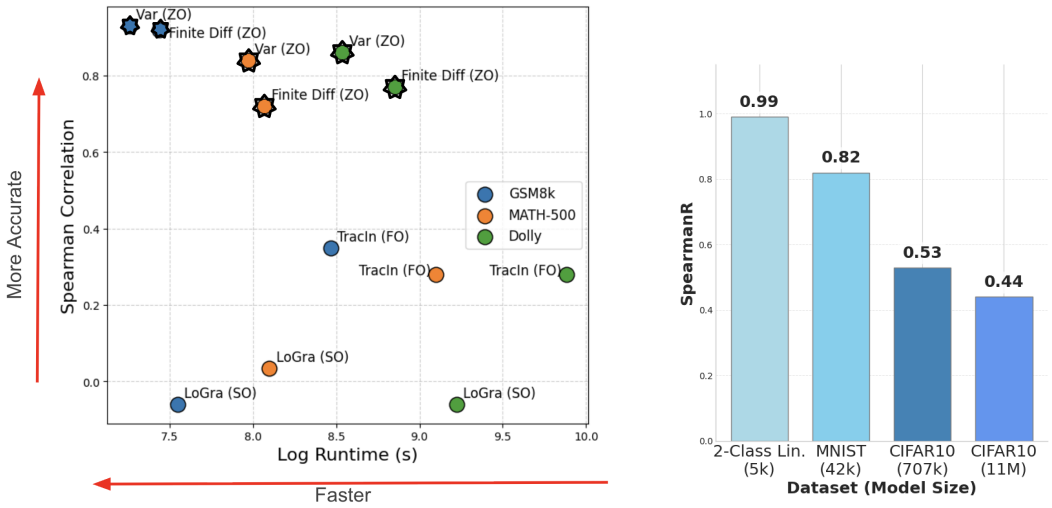

Figure 1: Spearman Correlation between SSRT and Data Influence approximation (accuracy) vs. runtime in Log(seconds) for four influence approximation methods for self-influence. Each color is a different LLM fine-tuning dataset. Our methods are encircled in star (**Left**) and Spearman correlation between the left and right hand sides of Equation 3 for different models sizes. The plot shows degradation of the gradient approximation with the increase of model size. (**Right**)

our method's computation cost as well as how accurately it approximates the "remove-and-retrain" baseline (Figure 1). We find that our method, due to its zeroth-order nature, offers *an order of magnitude* savings in both computation time and memory footprint especially for fine-tuned LLMs. We also find that it is more accurate than TracIn and SoTA data influence approximation techniques such as LoGra [Choe et al., 2024], especially for LLMs and self-influence. When correlated against the SubSample-and-Retrain (SSRT) influence estimator of Feldman and Zhang [2020], a slightly more computationally efficient version of the "remove-and-retrain" baseline, our method shows high correlations, which are higher than TracIn for fine-tuned LLMs.

In conclusion, by significantly reducing the computational overhead associated with data influence calculations, our method enables scalable and efficient analysis even in large models. This makes it feasible to integrate data influence into LLMs, enabling a number of real-world applications such

as model debugging, data attribution, data pruning, and training data auditing. We hope that this improved speed and accuracy will lead to more uses of self-influence in particular in modern LLMs, ultimately enabling higher model trustworthiness in domains like healthcare, finance, and AI safety.

1.1 Related Work

As we mentioned earlier, the algorithmic efficiency of the influence functions has improved over the years. In the inverse Hessian formulation, Schioppa et al. [2021] utilizes Arnoldi iteration to identify top-k eigenvectors and eigenvalues which then is used to approximate the inverse of Hessian. Grosse et al. [2023], Park et al. [2023], Kwon et al. [2023], Choe et al. [2024] focus on more advanced approximations of influence functions due to the increase of parameters in LLMs. Grosse et al. [2023] proposes Eigenvalue-corrected Kronecker-Factored approximate Curvature (EK-FAC) to approximate Inverse Hessian-Vector Product (IHVP) and scale influence calculations to 52B parameters. Kwon et al. [2023] on the other hand, focuses on LoRA fine-tuned models and proposes a faster closed-form approximation of IHVP. Lastly, Choe et al. [2024] proposes a more generic variant of Kwon et al. [2023] that applies both to pre-training and fine-tuning. It proposes a low-rank gradient projection algorithm, which significantly reduces the cost of IHVP estimation; we use this as one of our baselines. Lastly, Tu et al. [2025] replace the IHVP with ridge regression that avoids inverting the Hessian, nevertheless it still requires the loss to be twice differentiable.

Although efficient IHVP approximations are popular, they are still bounded by the assumptions and limitations described in Schioppa et al. [2023] and Li et al. [2024]. Pruthi et al. [2020], on the other hand, is based on the training trajectory of the model that is not bounded by these assumptions. It has been adopted by Xia et al. [2024] for selecting influential data points for instruction fine-tuned LLMs. Schioppa et al. [2023] argue, however, that many of those approximations are inaccurate due to lack of convexity, numeric stability and parameter divergence. Bae et al. [2024] addresses the concerns related to convexity using a segmented stationary unrolling technique, however, it still relies on first-order gradients. Yang et al. [2024] and Chhabra et al. [2025] simplify TracIn [Pruthi et al., 2020] further and show that only the gradients in the last checkpoint can be reliably used to identify detrimental training examples for certain applications; however, they still require first-order gradients. Ko et al. [2024] suffers from the same limitation. It proposes a reciprocal relationship between the train and the test and continuous model update on the test example. Another robust alternative, Ghorbani and Zou [2019], aims to approximate the remove-and-retrain baseline; however, it still requires model retraining. In a follow-up work Wang et al. [2024] addresses this limitation; however, it requires access to first- and second-order gradients.

2 Background

The influence function [Hampel et al., 1986] is a concept from robust statistics that helps us understand how small changes in the data can impact a statistical estimator, and by extension, a machine learning model trained on the data. In what follows, we assume that our training and test data are drawn from an underlying data distribution D, and training data $S = \{x_1, \ldots, x_n\}$ is used to build a model θ. Training is done by finding a model $\theta(S)$ that minimizes a loss function ℓ over the training data S. Specifically,

$$\theta(S) = \arg\min_{\theta} \sum_{i=1}^{n} \ell(\theta, x_i) \tag{1}$$

With this notation in mind, we can define the *true influence* of a training point x_s on the model's prediction on x_t as follows:

$$\inf(\ell, D, x_s, x_t) = \mathbf{E}_{S \sim D^n}[\ell(\theta(S \cup \{x_s\}), x_t) - \ell(\theta(S), x_t)] \tag{2}$$

Typically, influence values are computed for many training inputs x_s, usually the entire training set or a large sample thereof, to determine highly influential training data points for a particular test prediction.

Observe that true influence is extremely time-consuming to estimate even for the simplest of models: not only does it involve model training, which can be expensive by itself, but also a number of models need to be trained for accurate approximation. Consequently, a body of literature has looked into making the process more computationally tractable.

There are two avenues for improving the computation cost – first, reducing the number of models that need to be trained, and second, approximating the loss difference without fully re-training the model. In the first category, Feldman and Zhang [2020] proposes SSRT, where the idea is to retrain models on many subsamples of S of size $m < n$, and then use these retrained models to approximate Equation 2 together on many inputs x_s. Feldman and Zhang [2020] shows that this leads to an accurate approximation and hence we use this as our baseline for measuring accuracy.

However, even with this reduced number, model retraining remains expensive, particularly when it comes to more modern models. Consequently, two styles of approximations have been developed to estimate the loss difference $\ell(\theta(S \cup \{x_s\}), x_t) - \ell(\theta(S), x_t)$ itself.

The first style is a *second-order approximation* that was first proposed by [Koh and Liang, 2020], and is motivated by the fact that the impact of up-weighting a training point x_i on the loss minimizer $\theta(S)$ is approximately $-(\sum_j \nabla^2 \ell(\theta(S), x_j))^{-1} \nabla \ell(\theta(S), x_i)$. While this by itself is still expensive due to the Hessian computation, a body of work has been devoted to making it more algorithmically efficient. In particular, the more recent versions, such as LoGra, are quite competitive in terms of running time. However, the problem with this style of approximation is its accuracy: as has been observed in prior work [Li et al., 2024], as the approximations add up, the results depart significantly from SSRT – an observation that we corroborate as well.

The second style, that we call a *first-order approximation*, applies to models trained by gradient descent or its variants, and uses gradients at intermediate checkpoints to approximate the influence. The most popular among these is TracIn [Pruthi et al., 2020], which is motivated by the following theory. Suppose we have models θ_i and θ_{i+1} where θ_{i+1} is derived from θ_i by a single step of stochastic gradient descent. Then, for any data point x_t, we can approximate the difference in the losses on x_t by a first-order approximation as follows:

$$\ell(\theta_{i+1}, x_t) - \ell(\theta_i, x_t) \approx \nabla \ell(\theta_i, x_t)^\top (\theta_{i+1} - \theta_i) \tag{3}$$

In addition, since θ_{i+1} is obtained from θ_i in a single step of stochastic gradient descent, we have: $\theta_{i+1} - \theta_i = -\eta \nabla \ell(\theta_i, x_i)$, where η is the step size, and x_i is the training data point that was used in the update. Thus, while going from θ_i to θ_{i+1}, the influence of x_s on x_t is zero if x_s is not the updating point, and $-\eta \nabla \ell(\theta_i, x_s)^\top \nabla \ell(\theta_i, x_t)$ otherwise. The influence over the entire course of training can now be obtained by summing over all steps where x_s is used in the update:

$$\mathbf{TracIn}(x_s, x_t) = -\eta \sum_{i: x_i = x_s} \nabla \ell(\theta_i, x_s)^\top \nabla \ell(\theta_i, x_t) \tag{4}$$

Since $-\eta$ is a constant in all cases, we can remove it to get the final expression.

A major practical problem with Equation (4) is that its calculation requires access to the entire training trajectory, which is prohibitive to store. To address this, Pruthi et al. [2020] proposes a practical version where each θ_i is a checkpoint, usually obtained after an epoch of training. For the experiments in their paper, this practical version leads to both good accuracy and a reasonable running time.

However, how effective this approximation is for larger models still remains an open question. TracIn's utility hinges on the accuracy of the first-order approximation in Equation (3). To evaluate its accuracy, we measured the Spearman correlation between the left and right hand sides of this equation for consecutive model checkpoints θ_i and θ_{i+1} across models of different sizes (Figure 1). Our results reveal a key limitation: the approximation becomes less accurate as models grow larger and more complex. This suggests that Equation (3) provides an increasingly poor estimate of the actual difference in losses between checkpoints of large and complex models. At the same time, while gradients are cheaper for smaller models, they cost increasingly more as the size and complexity of the model increase. To remedy this, we instead directly look at finite differences in the losses, and compute a zeroth-order approximation that improves both the accuracy and the running time for large models. We describe the details in the next section.

3 Zeroth-order Approximation

Our numerical study in Figure 1 shows that the first order approximation in Equation 3 becomes increasingly loose with growing model size. At the same time, the gradient also becomes more expensive to calculate for the large models.

Motivated by this, we propose by-passing the gradient altogether, and directly fitting a vector to approximate the finite difference in the losses instead. Specifically, given two successive checkpoints θ_i and θ_{i+1} and a test point x_t, we aim to find a vector $g(\theta_i, x_t)$ such that $\ell(\theta_{i+1}, x_t) - \ell(\theta_i, x_t) \approx g(\theta_i, x_t)^\top(\theta_{i+1} - \theta_i)$; in other words, analogue to Equation (3), we hope to find a vector g that minimizes the following:

$$(\ell(\theta_{i+1}, x_t) - \ell(\theta_i, x_t) - g^T(\theta_{i+1} - \theta_i))^2. \tag{5}$$

Observe that Equation (5) has a closed form solution:

$$g(\theta_i, x_t) = \frac{(\ell(\theta_{i+1}, x_t) - \ell(\theta_i, x_t))(\theta_{i+1} - \theta_i)}{\|\theta_{i+1} - \theta_i\|^2}$$

$g(\theta_i, x_t)$, thus obtained, is also known as the finite difference directional gradient. It is a better first-order approximation to the loss difference between checkpoints by virtue of minimizing Equation (5); it is also faster to compute than a gradient since no backpropagation is required. To make it more reliable, we design a robust version that averages across a few closely related checkpoints: assuming that there are T checkpoints in all, and given a checkpoint θ_i let $M_i = \{j \in [T] \setminus \{i\}; d(\theta_i, \theta_j) \leq \tau\}$ be the set of indices of checkpoints that are within a certain distance τ of θ_i. Then, we can write:

$$\bar{g}(\theta_i, x_s) = \frac{1}{|M_i|} \sum_{j \in M_i} \frac{(\ell(\theta_j, x_s) - \ell(\theta_i, x_s))(\theta_j - \theta_i)}{\|\theta_j - \theta_i\|^2}. \tag{6}$$

Now, for a checkpoint θ_i, similar to TracIn, we approximate:

$$\ell(\theta_{i+1}, x_t) - \ell(\theta_i, x_t) \approx \bar{g}(\theta_i, x_t)^\top(\theta_{i+1} - \theta_i)$$

We then assume that a training point x_s changes the model by $\bar{g}(\theta_i, x_s)$; combining these approximations into a dot product and then summing over the checkpoints θ_i gives us the final solution. The full algorithm is described in Algorithm 1.

Algorithm 1 Zeroth-Order Influence

Input: A set of checkpoints $\theta_0, ..., \theta_T$, a training example x_s and a target example x_t. A distance metric d and a threshold τ.
Output: An estimate of the influence of x_s on x_t.
for $i \in \{0, \ldots, T\}$ **do**
  $M_i \leftarrow \{j \in [T] \setminus \{i\} : d(\theta_i, \theta_j) \leq \tau\}$
  $\bar{g}(\theta_i, x_s) \leftarrow \frac{1}{|M_i|} \sum_{j \in M_i} \frac{(\ell(\theta_j, x_s) - \ell(\theta_i, x_s))(\theta_j - \theta_i)}{\|\theta_j - \theta_i\|^2}$
  $\bar{g}(\theta_i, x_t) \leftarrow \frac{1}{|M_i|} \sum_{j \in M_i} \frac{(\ell(\theta_j, x_t) - \ell(\theta_i, x_t))(\theta_j - \theta_i)}{\|\theta_j - \theta_i\|^2}$
  $\mathbf{Zinf}(x_t, x_s, \theta_i) \leftarrow \langle \bar{g}(\theta_i, x_s), \bar{g}(\theta_i, x_t) \rangle$
end for
$\mathbf{Zinf}(x_t, x_s) \leftarrow \sum_{i=0}^{T} \mathbf{Zinf}(x_t, x_s, \theta_i)$
return: $\mathbf{Zinf}(x_t, x_s)$

3.1 Self-Influence: A Special Case

Self-influence is a special case of the influence function where we seek to find the impact of a training data point on itself; in other words, $x_s = x_t$. This is often used for detecting memorization, finding noise and outliers in training data.

It turns out that in the case of self-influence there is a simpler upper bound to the influence function value that can be computed even faster; we derive it next. Observe that for self-influence, we have:

$$\mathbf{Zinf}(x_s, x_s) = \sum_{i=1}^{T} \langle \bar{g}(\theta_i, x_s), \bar{g}(\theta_i, x_s) \rangle = \sum_{i=1}^{T} \|\bar{g}(\theta_i, x_s)\|^2 \tag{7}$$

For a specific θ_i,

$$\begin{aligned}\|\bar{g}(\theta_i, x_s)\|^2 &= \frac{1}{|M_i|^2}\Big\|\sum_{j\in M_i}\frac{(\ell(\theta_i,x_s)-\ell(\theta_j,x_s))(\theta_i-\theta_j)}{\|\theta_i-\theta_j\|^2}\Big\|^2 \\ &\le \frac{1}{|M_i|^2}\left(\sum_{j\in M_i}(\ell(\theta_i,x_s)-\ell(\theta_j,x_s))^2\right)\cdot\left(\sum_{j\in M_i}\frac{1}{\|\theta_i-\theta_j\|^2}\right), \\ &\le \left(\sum_{j\neq i}(\ell(\theta_i,x_s)-\ell(\theta_j,x_s))^2\right)\cdot\left(\sum_{j\neq i}\frac{1}{\|\theta_j-\theta_i\|^2}\right)\end{aligned}$$

Here, the second step follows from the Cauchy-Schwartz Inequality on vectors, and the third step follows because $|M_i| \geq 1$ and from the positivity of squares. If we define $c_i = \sum_{j\neq i}\frac{1}{\|\theta_j-\theta_i\|^2}$, and $c = \max(\{c_i; i \in [T]\})$ then the total self-influence becomes:

$$\mathbf{ZSinf}(x_s) = \sum_{i=1}^{T}\|\bar{g}(\theta_i, x_s)\|^2 \le c\cdot\sum_{i=1}^{T}\sum_{j\neq i}(\Delta\ell_{ij}(x_s))^2,$$

which is basically a constant times the variance of the losses $\ell(\theta_i, x_s)$ over the checkpoints θ_i. Thus, we see that the empirical variance of the losses on x_s is an upper bound on the self-influence, and we can use it as a proxy:

$$\mathbf{ZSinf}(x_s) = \mathbf{Var}_i(\ell(\theta_i, x_s)) \tag{8}$$

Appendix A.4 provides further analysis of the Cauchy-Schwartz slack based on the empirical evidence of the monotonic relationship between the ground truth and the variance of the loss.

One advantage of using the variance is that it is even faster to compute: unlike Algorithm 1, there is no need to compute the directions for the very high-dimensional models θ_i. The full algorithm is summarized in Algorithm 2. Appendix A.3 discusses further the accuracy of the zeroth-order approximation and its connection to TracIn.

Algorithm 2 Variance-based Self-Influence

1: **Input:** A set of checkpoints $\theta_0, ..., \theta_T$, a training example x_s.
2: **Output:** An estimate of the influence of x_s on itself.
3: $\mathbf{ZSinf}(x_s) \leftarrow \mathbf{Var}(\ell(\theta_0, x_s), \ldots, \ell(\theta_T, x_s))$
4: **return:** $\mathbf{ZSinf}(x_s)$

3.2 Optimizations for Higher Efficiency

We next describe two additional optimizations for computing $\mathbf{Zinf}$ and $\mathbf{ZSinf}$ even faster.

Dot Product Expansions. The first is to use the dot-product expansion rule over sums to further optimize the calculation of $\mathbf{Zinf}$. In particular, if we let $r_{ij}(x) = \ell(\theta_j, x) - \ell(\theta_i, x)$, then we can simplify the dot product $\langle\bar{g}(\theta_i, x_s), \bar{g}(\theta_i, x_t)\rangle$ as follows:

$$\begin{aligned}&\Big\langle\frac{1}{|M_i|}\sum_{j\in M_i}\frac{r_{ij}(x_s)(\theta_j-\theta_i)}{\|\theta_j-\theta_i\|^2}, \frac{1}{|M_i|}\sum_{j\in M_i}\frac{r_{ij}(x_t)(\theta_j-\theta_i)}{\|\theta_j-\theta_i\|^2}\Big\rangle \\ &= \frac{1}{|M_i|^2}\sum_{j\in M_i}\sum_{k\in M_i}\frac{r_{ij}(x_s)r_{ik}(x_t)(\langle\theta_j,\theta_k\rangle - \langle\theta_i,\theta_j\rangle - \langle\theta_i,\theta_k\rangle + \langle\theta_i,\theta_i\rangle)}{\|\theta_j-\theta_i\|^2\cdot\|\theta_k-\theta_i\|^2}.\end{aligned}$$

Observe that $(\langle\theta_j,\theta_k\rangle - \langle\theta_i,\theta_j\rangle - \langle\theta_i,\theta_k\rangle + \langle\theta_i,\theta_i\rangle)$ are byproducts of the Gramm matrix of weights which we can pre-compute once and reuse for all x_s and x_t. The same applies to the norms $\|\theta_j-\theta_i\|^2\cdot\|\theta_k-\theta_i\|^2$; this optimization significantly accelerates our method for train-test influence.

In addition to the Gramm matrices of the θ_i's, we can also do offline pre-computation, where we compute and store the loss of each checkpoint over the data during training and reuse during the train-test influence evaluation. We report the offline and online runtime of our method in Figure 13.

4 Experiments

4.1 Experimental Setup

Datasets We used MNIST [LeCun et al., 2010] and CIFAR10 [Krizhevsky, 2009] datasets for our vision models, GSM8K [Cobbe et al., 2021], MATH-500 [Hendrycks et al., 2021], Dolly [Conover et al., 2023] and WinoGrande [Keisuke et al., 2019] to train our LLMs. Additionally, we utilized a linearly separable 2-class classification synthetic dataset (20k examples, 5k features, drawn uniformly random).

Models We trained a linear model on our synthetic 2-class classification dataset using Adam optimizer with a learning rate of 0.001 for 100 epochs. Additionally, we trained a simple ConvNet on MNIST [PyTorch Contributors, 2024] with learning rate 0.001 and Deep Aggregate ConvNet [Yu et al., 2018] on CIFAR10 dataset with learning rate 0.001 and weight decay 5e-4. We fully fine-tuned LLama3.1 8B [Team, 2024] using Fairseq2 [Balioglu et al., 2023] on GSM8K, MATH-500 and Dolly datasets using weighted Adam optimizer. We made adjustments in TracIn similar to Xia et al. [2024] to account for optimizer's parameter update intricacies, since our LLMs use the Weighted Adam optimizer with a learning rate 5.5e-06. Our experiments, however, suggest that these adjustments make little or no difference in the TracIn or zeroth-order approximation.

SubSample and Retrain Setup We ran 100 trials of the subsample and retrain experiments on a 0.7 fraction of the training data sampled uniformly random from the training data distribution, similar to Feldman and Zhang [2020]. We applied this setup to all models and all experiments.

Influence Approximation Setup We used only 10 to 15 early epochs for the zeroth-order approximation and TracIn approximation. We estimate zeroth-order using two techniques: 1)Variance of the loss across all checkpoints (ZSinf) and 2) Finite differences gradient approximation (Zinf) described in Section 3. We only used last layer parameters for TracIn and zeroth-order (Zinf) without random projections for fair comparison since TracIn for all parameters becomes computationally prohibitive for LLMs. We used all layer parameters both for TracIn and zeroth-order (Zinf) for non-LLM models trained on synthetic dataset, CIFAR and MNIST. We use rank 64 for low rank gradient projection in LoGra. A rank larger than that leads to OOM on a single A100 machine with 8 GPU nodes.

Evaluation Metrics We use Spearman Rank Correlation [Spearman, 1987] metric between SSRT and influence approximation methods both for self and train-test influence. We also considered the Linear Data Modeling Score (LDS) [Park et al., 2023] metric, however, the metric is noisy and requires model retraining which is not feasible for large models. We report results on LDS in Appendix A.5. We measured the runtime performance on a single A100 machine with 8 Cores.

4.2 Accuracy of the SSRT Ground Truth

We validate Subsample-and-Retrain (SSRT) as a proxy for ground truth in two ways. First, for 100-trial SSRT, we split the trials into two independent sets of 50 and compute the Spearman correlation between the two halves (Figure 2). The reliability ordering (Dolly > MATH-500 > GSM8k) tracks the heterogeneity of the datasets rather than their size. The split-half reliability (from 0.754 to 0.842) proves SSRT measures something real and reproducible at LLM scale. These reliabilities establish an upper bound on the correlation any method can achieve with SSRT bounded by the Spearman-Brown [Brown, 1910] corrected reliability (GSM8k (0.86), MATH-500) (0.87), Dolly (0.91)).

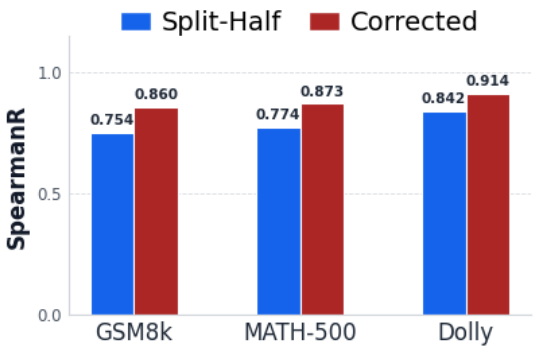

Figure 2: SSRT split-half and Spearman-Brown corrected reliability.

Second, in a small linear model where exact LOO influence is computable, SSRT and exact LOO influence are strongly correlated across 100 training examples (Spearman $\rho = 0.869$, $p = 9.0 \times 10^{-32}$). This provides direct convergent evidence that SSRT approximates the remove-and-retrain in a setting where ground truth is available.

4.3 Correlation Accuracy and Runtime Analysis

In this section, we discuss how accurately SSRT correlates with zeroth, first, and second-order approximation methods. Figure 1 demonstrates that our method is approximately 3 times faster and more accurate than TracIn. LoGra has comparable runtime, however, it exhibits almost no correlation with the ground truth, SSRT. Appendix A.9 demonstrates additional results on vision and simple linear models. It shows that all influence approximation methods perform comparable when it comes to the correlation accuracy in linear and simple vision models, however, their performance declines significantly as complexity increases.

Furthermore, we find that approximately 80 to 90% of the runtime of our method is devoted to computing the loss for train and test examples in each epoch. We note that the loss for train examples can be precomputed once for all epochs and then reused during evaluation, reducing redundant computation. We also observe that the first 5–10 epochs are critical for achieving accurate approximations, while subsequent epochs have less impact on accuracy especially for LLMs (Figure 5).

In our experiments, we also study the common subset of top-k influential training examples between the SSRT and self-influence approximations. Figure 3 shows that zeroth-order has significantly greater

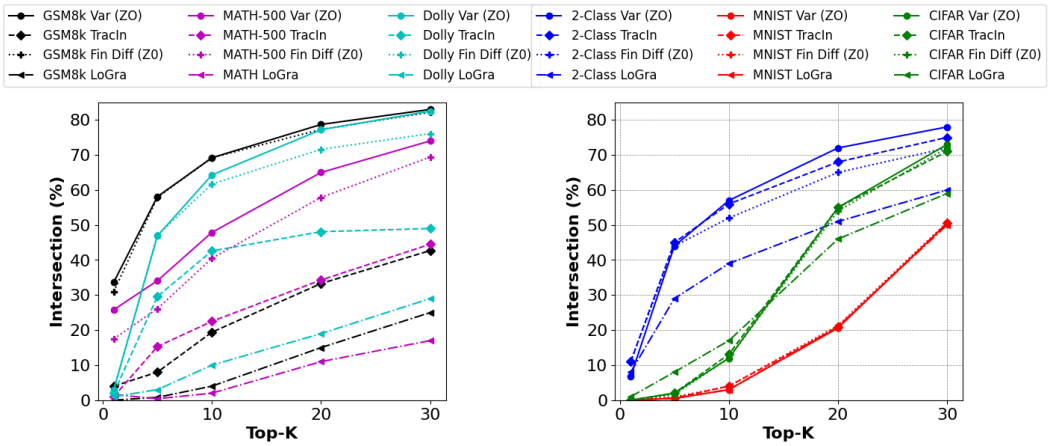

Figure 3: The common subset of top-k self-influential examples between SSRT and four different influence approximation methods for three fine-tuned LLMs (**Left**) and non-LLMs (**Right**) in %.

overlap with SSRT, resulting in a larger area under the curve for LLMs. However, small models trained on CIFAR, MNIST and 2-class classification synthetic data (Figure 3), show comparable results across first- and zeroth-order methods. LoGra consistently underperforms across all types of models. Our findings indicate that the performance of our method improves as the size of the model increases, consistent with the theory discussed in Section 2.

4.4 Checkpoint, Neighborhood, Optimizer and Parameter Selection

In this section we study the impact of checkpoints, neighborhoods, optimizers, and model size on SSRT correlation.

Phase-wise view. Figure 5 visualizes six phases of model training, each phase grouped in 10 checkpoints. The checkpoints in each phase are used to estimate first- and zeroth-order approximations. The plot reveals that the gap between zeroth- and first-order methods is the largest in the early 10 epochs. This is also reflected by the variance of the average loss across all 10 epochs in each phase. Appendix A.7 provides a further breakdown of the checkpoint-wise view. **Neighborhood.** Figure 7 shows increasing the number of neighbors used by zeroth-order improves the quality of approximation but overly distant



Figure 4: Correlation between checkpoint index difference and parameter distance. For each pair of checkpoints (i, j), $|i - j|$ measures the difference in checkpoint indices, and $\|\theta_i - \theta_j\|_2$ measures the L_2 distance between their parameters.

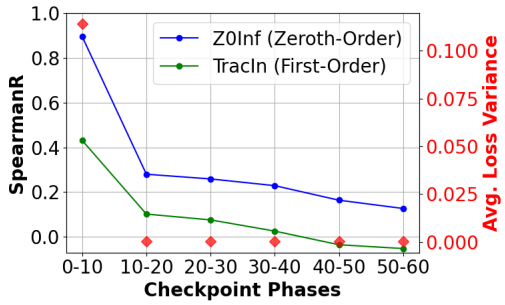

Figure 5: Phase-wise view of the correlation accuracy along with the average variance of the loss for Dolly dataset.

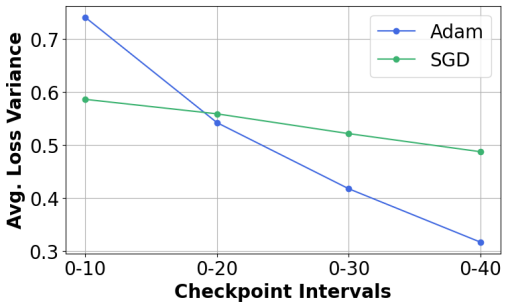

Figure 6: Variance of average loss over checkpoint intervals for ConvNet.

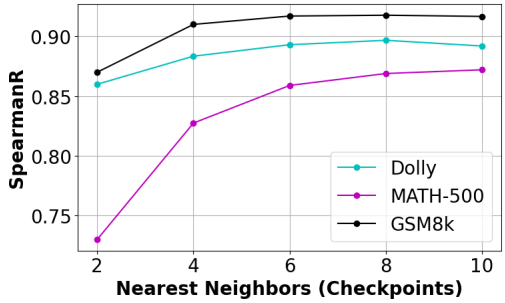

Figure 7: The impact of the number of nearest neighbors on the accuracy of the zeroth-order for 10 epochs.

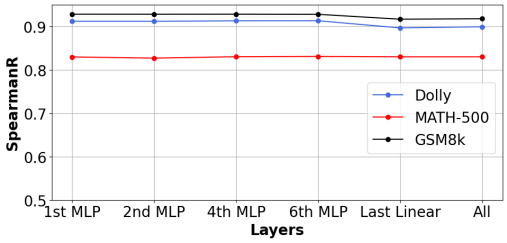

Figure 8: SpearmanR between SSRT and Self-Influence (Zeroth-Order) for four MLP, last linear layer and all model parameters, trained for 10 epochs.

neighbors can introduce noise. We use the adjacency index as the nearest-neighbor metric, which is highly correlated with L2 distance as depicted in Figure 4.

Optimizer sensitivity. Figure 6 shows that SGD exhibits a slower, more linear loss decay compared to Adam. Moreover, relative to LLM fine-tuning (Figure 5), ConvNet convergence is more gradual.

Likely, because fine-tuning adjusts pre-trained weights near a favorable minimum, whereas ConvNets optimize from random initialization. Additional SGD vs. Adam results are provided in Appendix A.8.

Parameter Selection. Figure 8 shows that correlation accuracy is almost insensitive to layer selection. This explains why the loss variance alone achieves high correlation accuracy in Section 3.1.

4.5 Train-Test Influence

In addition to self-influence, we also study train-test influence on several models and datasets. We apply only the finite differences method to approximate zeroth-order since variance-based approximation is not fully generalizable to train-test scenarios. We first contaminate the training datasets by introducing WinoGrande samples, resulting in contamination ratios of 13% for Dolly, 25% for GSM8k, and 17% for MATH-500. We then measure the proportion of the most influential training examples that originate from the same distribution as the test set (i.e. non-WinoGrande). Figure 9 reveals that our zeroth-order method offers significant speedup compared to TracIn and achieves similar or better performance in data selection. While LoGra is faster than TracIn, it achieves lower accuracy. As a baseline, we evaluated random data selection averaged over five independent trials, obtaining accuracies of 80% on MATH-500, 84% on Dolly, and 74% on GSM8k.

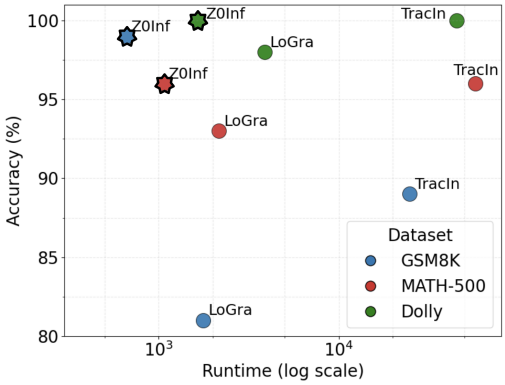

Figure 9: Data selection accuracy vs. Runtime in Log(seconds). The accuracy is evaluated for top 2% of most influential training examples.

The consistent improvement of influence-based selection over this baseline confirms the effectiveness of data influence approximation methods. Furthermore, we observe that the SSRT measure of train-test influence is semantically different from existing first- and second-order influence approximation methods. While exact gradient-based approximation methods measure the influence of a training example on the test as a similarity between gradients, SSRT quantifies the influence of a training example on the test as a total effect of sample elimination. Hence, it does not rely on input or gradient similarity.

SSRT excels at finding train-test pairs that are from the same distribution and unique near duplicates [Feldman and Zhang, 2020] of each other. If there are multiple similar examples in the training set, then the influence of a single example from that group on a test example is insignificant. Thus, even if the train-test pair is "visually" similar, the influence might still be low due to the other similar examples in the training set. First- and second-order methods depend on exact gradients, which often include activations as a multiplicative factor (Appendix A.6). Hence, existing influence approximations tend to find training examples that are similar to test ones while having less correlation with SSRT. Our method, however, does not suffer from this limitation since Equation 6 relies purely on evaluation loss and model weights. While zeroth-order does not directly incorporate the representation similarity between the train and test, it still captures unique train-test pairs that are near duplicates similar to SSRT. Table 2 demonstrate such examples for Dolly datasets and Figure 16, further demonstrates the effectiveness of our approach for CIFAR10 vision dataset.

5 Discussion

Our experiments show that even with 100 training trials (models), the zeroth-order approximation for self-influence correlates strongly with SSRT. We conjecture that since self-influence represents the gradient norm, unlike a similarity metric between train and test, it circumvents the challenges of sample-level interactions. In other words, if a training example has high self-influence because it is an outlier, other examples in the training set cannot influence it. In contrast, train-test is only correlated for a subset of influential top-k examples with SSRT (Appendix A.10). We observe that only a small subset of training examples exerts a disproportionate influence on test-time predictions. Furthermore, for a subset of test examples, no clearly distinguishable set of top-k most influential training examples emerges. In the future, we plan to look deeper into characterizing these influence dynamics, example-level interactions and subset (group) influence. We believe that group influence will provide higher correlation with SSRT since in this setting we group similar training examples together. The confounding effects caused due to the example-level interaction and correlations will be further alleviated in this setting. Furthermore, we plan to evaluate the zeroth-order approximation on downstream tasks, for data selection and safety applications.

6 Conclusion

This work proposes a zeroth-order influence approximation algorithm that offers both substantial performance and accuracy improvements. We provide theoretical analysis establishing its connection to first-order approximations and demonstrate empirically that it attains significantly higher correlation with SSRT for self-influence while proving effective for train-test influence estimation. We validate our approach across diverse vision and language models and discuss the limitations and strengths of SSRT and influence approximation methods.

Acknowledgments and Disclosure of Funding

We sincerely thank Ilia Kulikov for his help setting up the fine-tuning pipelines with Fairseq2 and for answering our related questions.

This work was conducted at Meta, and all authors were employees of Meta Platforms, Inc. at the time. Kamalika Chaudhuri and Saeed Mahloujifar are now employed by Google DeepMind. The work was supported through the authors' employment at Meta and received no third-party funding.

References

Juhan Bae, Wu Lin, Jonathan Lorraine, and Roger Grosse. Training data attribution via approximate unrolled differentiation. 2024. URL https://arxiv.org/abs/2405.12186.

Can Balioglu, Martin Gleize, Artyom Kozhevnikov, Ilia Kulikov, Tuan Tran, and Julien Yao. fairseq2, 2023. URL http://github.com/facebookresearch/fairseq2.

Samyadeep Basu, Philip Pope, and Soheil Feizi. Influence functions in deep learning are fragile, 2021. URL https://arxiv.org/abs/2006.14651.

William Brown. Some experimental results in the correlation of mental abilities. *British Journal of Psychology*, 3(3):296–322, Oct 1910. ISSN 0950-5652. doi: 10.1111/j.2044-8295.1910.tb00207.x. URL http://dx.doi.org/10.1111/j.2044-8295.1910.tb00207.x.

Anshuman Chhabra, Bo Li, Jian Chen, Prasant Mohapatra, and Hongfu Liu. Outlier gradient analysis: Efficiently identifying detrimental training samples for deep learning models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=v77ZMzbsBA.

Sang Keun Choe, Hwijeen Ahn, Juhan Bae, Kewen Zhao, Minsoo Kang, Youngseog Chung, Adithya Pratapa, Willie Neiswanger, Emma Strubell, Teruko Mitamura, Jeff Schneider, Eduard Hovy, Roger Grosse, and Eric Xing. What is your data worth to gpt? llm-scale data valuation with influence functions, 2024. URL https://arxiv.org/abs/2405.13954.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

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.

Junwei Deng, Ting-Wei Li, Shiyuan Zhang, Shixuan Liu, Yijun Pan, Hao Huang, Xinhe Wang, Pingbang Hu, Xingjian Zhang, and Jiaqi Ma. A library for efficient data attribution. 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 136763–136781. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/f732683302d91e47610b2416b4977a66-Paper-Datasets_and_Benchmarks_Track.pdf.

Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: discovering the long tail via influence estimation. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

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.

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.

Frank Hampel, Elvezio Ronchetti, Peter J. Rousseeuw, and Werner A. Stahel. Robust statistics: the approach based on influence functions. 1986. URL https://api.semanticscholar.org/CorpusID:209831924.

Satoshi Hara, Atsushi Nitanda, and Takanori Maehara. Data cleansing for models trained with sgd. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/5f14615696649541a025d3d0f8e0447f-Paper.pdf.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021.

Karthikeyan K and Anders Søgaard. Revisiting methods for finding influential examples, 2021. URL https://arxiv.org/abs/2111.04683.

Sakaguchi Keisuke, Le Bras Ronan, Bhagavatula Chandra, and Choi Yejin. Winogrande: An adversarial winograd schema challenge at scale. 2019.

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.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions, 2020. URL https://arxiv.org/abs/1703.04730.

Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

Yongchan Kwon, Eric Wu, Kevin Wu, and James Zou. Datainf: Efficiently estimating data influence in lora-tuned llms and diffusion models. *arXiv preprint arXiv:2310.00902*, 2023.

Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Zhe Li, Wei Zhao, Yige Li, and Jun Sun. Do influence functions work on large language models?, 2024. URL https://arxiv.org/abs/2409.19998.

Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. Trak: attributing model behavior at scale. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 19920–19930. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/e6385d39ec9394f2f3a354d9d2b88eec-Paper.pdf.

PyTorch Contributors. Defining a neural network — pytorch tutorials. https://pytorch.org/tutorials/recipes/recipes/defining_a_neural_network.html#specify-how-data-will-pass-through-your-model, 2024. Accessed: 2024-06-10.

Andrea Schioppa, Polina Zablotskaia, David Vilar, and Artem Sokolov. Scaling up influence functions, 2021. URL https://arxiv.org/abs/2112.03052.

Andrea Schioppa, Katja Filippova, Ivan Titov, and Polina Zablotskaia. Theoretical and practical perspectives on what influence functions do. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Charles Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 100(3/4):441–471, 1987.

Llama Team. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024. doi: 10.48550/ARXIV.2407.21783. URL https://doi.org/10.48550/arXiv.2407.21783.

Zhuozhuo Tu, Cheng Chen, and Yuxuan Du. RRInf: Efficient influence function estimation via ridge regression for large language models and text-to-image diffusion models. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 18494–18507, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.933. URL https://aclanthology.org/2025.emnlp-main.933/.

Jiachen T Wang, Prateek Mittal, Dawn Song, and Ruoxi Jia. Data shapley in one training run. *arXiv preprint arXiv:2406.11011*, 2024.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: Selecting influential data for targeted instruction tuning. In *International Conference on Machine Learning (ICML)*, 2024.

Ziao Yang, Han Yue, Jian Chen, and Hongfu Liu. Revisit, extend, and enhance hessian-free influence functions, 2024. URL `https://arxiv.org/abs/2405.17490`.

Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2403–2412, 2018.

A Appendix

A.1 Broader Impact

This work presents a scalable and efficient algorithm for identifying the most influential training examples in machine learning systems. The proposed method facilitates model transparency, enables data valuation and outlier detection during model training, and is applicable to both LLMs and traditional machine learning models. The primary goal is to increase model transparency and assess the quality of the data during training.

There are no negative societal impacts of the work performed in this paper.

A.2 License of the assets

License for the models We use LLama3.1 8B with Llama 3.1 Community License and Fairseq2 [Balioglu et al., 2023] framework with MIT License.

License for the datasets The language datasets GSM8k [Cobbe et al., 2021] and MATH-500 [Hendrycks et al., 2021] are under MIT License, Dolly [Conover et al., 2023] and Winogrande [Keisuke et al., 2019] datasets under CC BY-SA 3.0 and CC-BY respectively.

The vision datasets CIFAR-10 [Krizhevsky, 2009] and MNIST [LeCun et al., 2010] are licensed under MNIST under MIT license and Creative Commons Attribution-Share Alike 3.0 license respectively.

A.3 Zeroth-order Approximation Accuracy

TracIn and Z0-Inf are based on the same formulation of loss change described in Equation 3. They differ only in which side of the equation they approximate.

Let $d_i = \theta_{i+1} - \theta_i$ and $\Delta\ell_i(x) = \ell(\theta_{i+1}, x) - \ell(\theta_i, x)$. TracIn does not measure $\Delta\ell_i(x)$, it approximates the loss change from the gradient of the first-order Taylor expansion.

$$\Delta\ell_i(x) \approx \nabla\ell(\theta_i, x)^\top d_i \tag{9}$$

This approximation assumes the loss surface is a flat plane between θ_i and θ_{i+1}. The real surface, however, is curved, and the error of the approximation is the second-order term:

$$\text{error} = \tfrac{1}{2}\, d_i^\top \nabla^2\ell(\xi, x)\, d_i \quad \text{for some } \xi \text{ between } \theta_i \text{ and } \theta_{i+1}$$

This error is small only when the step is small and the surface is close to flat. Large model training breaks both conditions. Figure 1 shows that as models get larger, TracIn's predicted loss change moves further from the actual loss change. Z0-Inf, on the other hand, measures the loss change, it doesn't approximate it. It, however, approximates the gradient $g(\theta_i, x)$ from the closed form solution based on the Equation 5.

$$g(\theta_i, x) = \frac{\Delta\ell_i(x)}{\|d_i\|^2}\, d_i$$

It holds for any optimizer, any step size and any amount of curvature, because nothing is expanded or truncated.

Z0-Inf, however, has other shortcomings. Because g points along d_i, Z0-Inf only sees how the loss changes along the direction training actually moves. Gradient components perpendicular to the training direction (more precisely, to the span of the neighbor displacements $\{\theta_j - \theta_i\}_{j \in M_i}$) are dropped. Dropping the perpendicular components therefore costs much less than an error in the loss change itself. For self-influence ($x_s = x_t$) only the size of the change matters, the direction plays no role.

A.4 Cauchy-Schwarz Slack

In this section we empirically quantify Cauchy-Schwarz slack by plotting (Figure 10) the relationship between the variance of the loss across early checkpoints and the SSRT ground truth. The plot shows clear monotonicity of the variance as the SSRT increases.

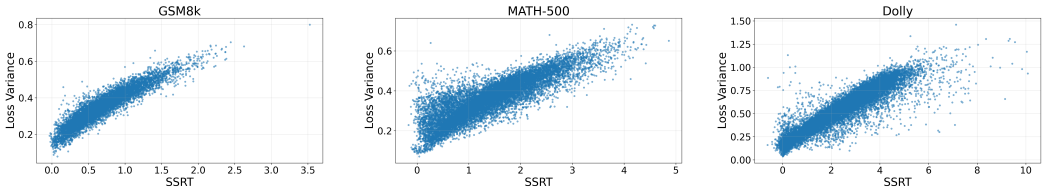

Figure 10: Loss Variance vs SSRT for three fine-tuning datasets. The variance is computed for the first 10 checkpoints for GSM8K and Dolly datasets and for the first 30 checkpoint for the MATH-500 dataset.

A.5 Linear Data Modeling Score

Similar to the prior work Park et al. [2023], Choe et al. [2024], Deng et al. [2024] we use Linear Data Modeling Score (LDS) to measure the quality of train-test influence. Our experiments reveal that LDS is a noisy metric that does not fully capture remove and retrain effects. The stability of the LDS score depends on the number of models trained on a training subset with random parameter initializations. Generally, it results in a large average p-value score which does not support the statistical significance of those results. Table 1 shows that while the LDS score is very similar for the Convnet model, it quickly starts to worsen significantly for TracIn and LoGra methods when applied to LLM.

	CIFAR-10	**GSM8k**
TracIn	0.02	0.0003
Zeroth Order	0.021	0.009
LoGra	0.026	-0.003

Table 1: LDS for two datasets and three approximation methods. As the models become larger, Zeroth-Order shows slightly larger correlation compared to First- and Second-Order approximations.

A.6 SSRT vs Data Influence Approximations

We show that zeroth-order gradient approximation can be effective for large models, however, there are also fundamental differences between exact and approximated gradients. These differences can play an important role in estimating train-test influence. Below we demonstrate those differences on an example of a linear model. Let's assume that we are given the following linear model $\hat{y} = \sigma(W^T x)$ where $x \in \mathbb{R}^d$ is the input, $W \in \mathbb{R}^d$ is the weight matrix and $\hat{y}$ is the prediction scalar. Given the cross-entropy loss $L(W) = -[ylog(\hat{y}) + (1-y)log(1-\hat{y})]$, the exact gradient for the linear model can be formulated as follows: $\nabla_W L = (\hat{y} - y)(x)$ On the other hand, the zeroth-order approximation 6 does not have the multiplier with input x. This explains why most influential train-test pairs are the most similar, yet do not always correlate with the SSRT estimates. From this point of view zeroth-order approximation provides stronger signal that does not rely on input similarities.

A.7 Checkpoint selection

In this section we provide additional checkpoint-wise experimental results both for TracIn and Zeroth-Order along with the loss change. We observe that the correlation with SSRT slowly degrades (Figure 11) both for Zeroth-Order and TracIn, however, the gap in the beginning is slightly larger.

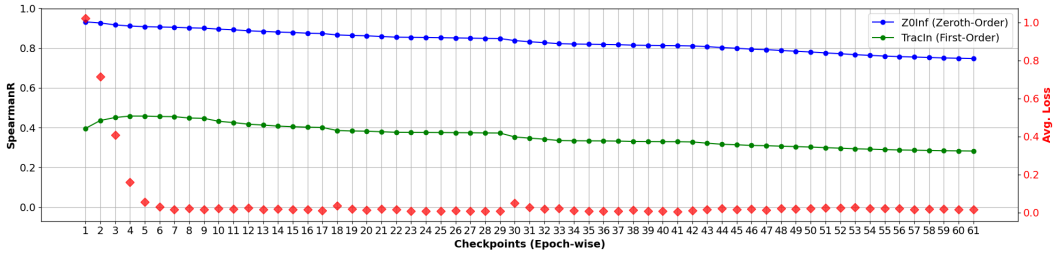

Figure 11: Epoch-wise loss and correlation accuracy change both up to given checkpoint for TracIn and Zeroth-Order for Dolly dataset.For a given checkpoint, TracIn and Zeroth-Order scores are computed using all training steps up to that checkpoint. Average loss represents the average loss across all training examples at given checkpoint.

A.8 SGD vs Adam

Figure 12 demonstrates how the Spearman correlation changes as we increase the number of checkpoints used in the first- and zero-order approximation. It is interesting to observe that the correlation is much higher for Adam compared to SGD. We assume that since SGD takes much longer to converge the correlation gap between SGD and Adam is especially large in the first 10 checkpoints. Figure 12 also reveals that the gap between Z0-Inf and TracIn is slightly larger for Adam optimizer.

A.9 Accuracy vs Runtime Analysis for Self-Influence

Figure 13 presents a comprehensive analysis of runtime and correlation accuracy across three fine-tuned language models, two vision models, and one linear model. The results indicate that the zeroth-order method achieves performance comparable to first- and second-order methods on smaller models and substantially surpasses other approaches when applied to large language models (LLMs). Additionally, the figure highlights that a significant portion of the total runtime is dedicated to computing the loss and GRAM matrices, whereas the actual zeroth-order computation accounts for only a small fraction of the overall time.

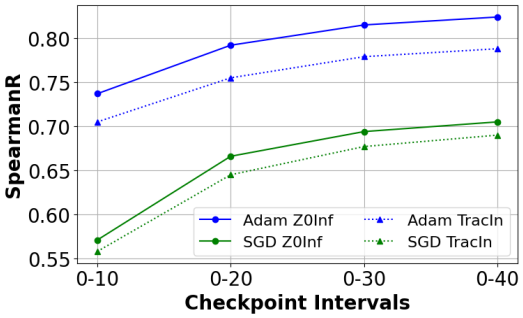

Figure 12: SpearmanR between SSRT and influence approximation methods for SGD vs Adam using CIFAR10 dataset.

A.10 Train-Test Data Influence Examples

Figure 14 shows that the correlation between SSRT and influence approximation methods is much smaller for train-test compared to self-influence. We believe that many training examples have unremarkable influence on test, resulting in noisier estimates and lower correlation. Figure 14 showcases that zeroth-order outperforms LoGra and TracIn on Dolly and MATH-500 datasets and slightly underperforms TracIn on GSM8k dataset. A slightly higher correlation score for TracIn may be attributed to a high degree of input resemblance between certain train and test examples, which in turn makes those training examples particularly influential for the corresponding test examples. Overall, LoGra performs poorly on all three datasets.

In addition, Table 2 presents most influential train-test pairs for Dolly dataset. It shows that the most influential pairs are semantically similar and are not purely based on the shared amount of tokens.

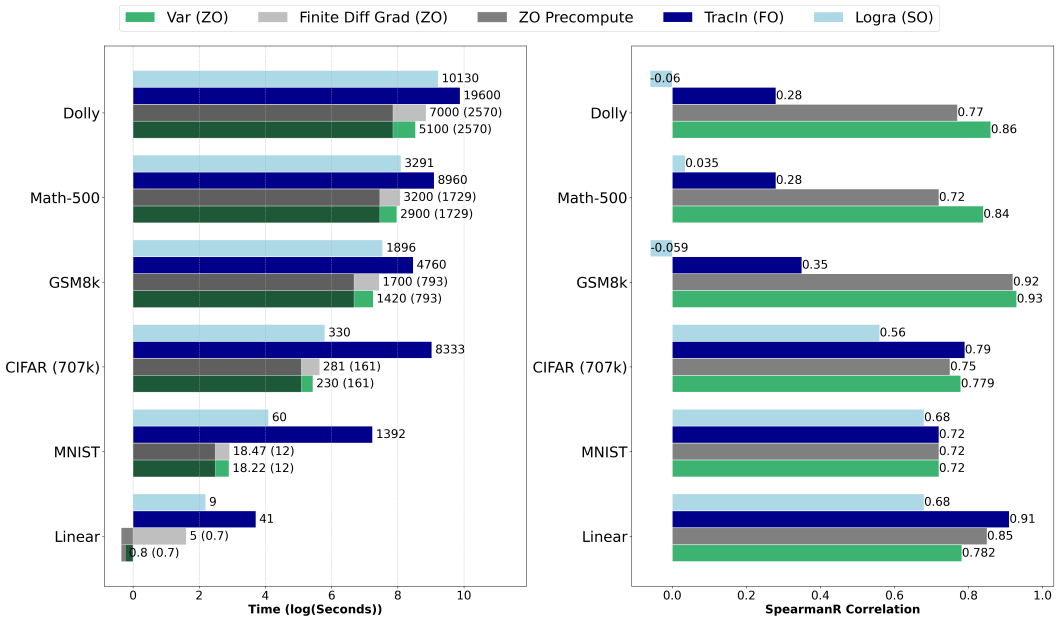

Figure 13: Left: Runtime performance comparison across different optimization methods for six different models. The offline precompute required for our method is overlayed in dark gray. Right: Spearman rank correlation coefficient between SSRT, LoGra and First and Zeroth-order approximation methods.

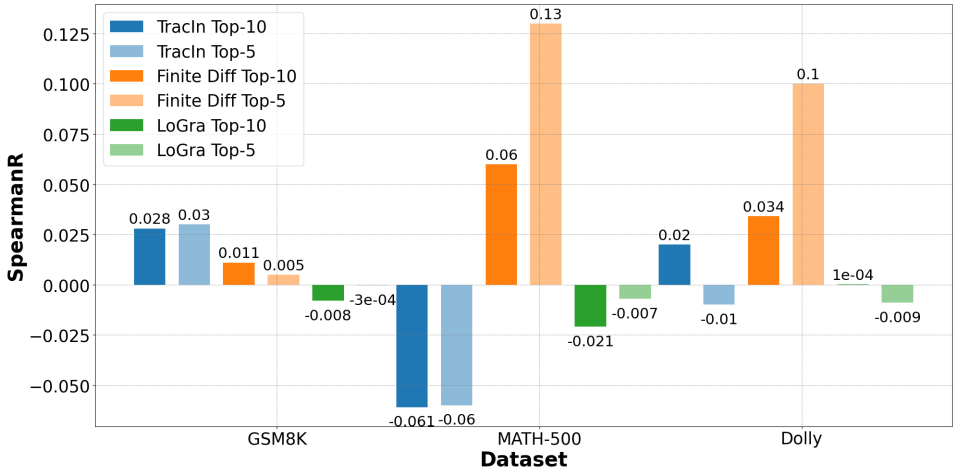

Figure 14: Average Spearman correlation between TracIn, Zeroth-Order, LoGra and SSRT for Top-5 and Top-10 most influential training examples. The training examples are selected based on the largest aggregate influence over all test examples for each method.

Similarly, Figure 15 and Figure 16 show that we are able to find unique near duplicates in the training data that are very influential for a very similar test example.

Table 2: Top-3 most influential train-test pairs based on SSRT and influence approximaiton methods for Dolly dataset. Rank denotes the influence rank of the training example relative to the specific test example.

Test	Train	Rank

Q: Below is an instruction that describes a task. Write a response that appropriately completes the request.### Instruction: Think up some openings for beginners to the game of Chess. Input: A: Chess openings are the most important part of the game, there are some advanced ones, some common ones, and some easy ones. This is a list of openings for beginners to the game:- The King's Pawn Opening: This is a very common opening, you move your king's pawn 2 spaces forward, so does the opponent. This is followed by developing your knights. - The Fianchetto defense. You move your kingside knight's pawn forward 1 space, then you develop the kingside knight. You then put your kingside bishop in the square that the pawn used to be in, followed by castling. This opening is common among lower ranked players. - Four Knights Game: This doesn't happen too much. Both sides start by developing their knights, the pawns take it over from there.- The French Defense: You move your king's pawn 1 space forward, followed by developing all of your kingside pieces, and finishing by castling.	Q: Below is an instruction that describes a task. Write a response that appropriately completes the request.### Instruction:Think up some openings for beginners to the game of Chess. Input: A: Chess openings are the most important part of the game, there are some advanced ones, some common ones, and some easy ones. This is a list of openings for beginners to the game: - The King's Pawn Opening: This is a very common opening, you move your king's pawn 2 spaces forward, so does the opponent. This is followed by developing your knights. - The Fianchetto defense. You move your kingside knight's pawn forward 1 space, then you develop the kingside knight. You then put your kingside bishop in the square that the pawn used to be in, followed by castling. This opening is common among lower ranked players. - Four Knights Game: This doesn't happen too much. Both sides start by developing their knights, the pawns take it over from there. - The French Defense: You move your king's pawn 1 space forward, followed by developing all of your kingside pieces, and finishing by castling.	SSRT: 1, ZInf: 4, TracIn: 26, LoGra: 489
Q: Below is an instruction that describes a task. Write a response that appropriately completes the request.### Instruction:How can I decide what bike I should get? Input: A: When it comes to bicycles for adults, there are different kinds available in the market. Select a bike based on how you'd like to use it. Bikes can be categorized as follows:1. Road bikes - These bikes have super thin tires and are often used for biking on roads. These help you attain high speeds relatively quickly.2. Mountain bikes - These bikes are made to sustain the rough terrain that you could come across while biking. The tires are much thicker made to protect you and give you the support you need.3. Hybrid/Commuter bikes - These bikes fall into the middle of the above 2 categories. If you think you might do a bit of road biking and mountain biking, then this might be your best option.You can also go to a store that specializes in selling and repairing bikes to get more information.	Q: Below is an instruction that describes a task. Write a response that appropriately completes the request.### Instruction:How can I decide what bike I should get? Input: A: When it comes to bicycles for adults, there are different kinds available in the market. Select a bike based on how you'd like to use it. Bikes can be categorized as follows:1. Road bikes - These bikes have super thin tires and are often used for biking on roads. These help you attain high speeds relatively quickly.2. Mountain bikes - These bikes are made to sustain the rough terrain that you could come across while biking. The tires are much thicker and made to protect you and give you the support you need.3. Hybrid/Commuter bikes - These bikes fall into the middle of the above 2 categories. If you think you might do a bit of road biking and mountain biking, then this might be your best option. You can also go to a store that specializes in selling and repairing bikes to get more information.	SSRT: 1 ZInf: 1 TracIn: 1 LoGra: 780

Q: Below is an instruction that describes a task. Write a response that appropriately completes the request.### Instruction: Best film in Motion Picture 27th Screen Actors Guild(SAG) Awards was given to which film? Input: A: Chicago 7	Q: Below is an instruction that describes a task. Write a response that appropriately completes the request.### Instruction: Give me a list of scientists who influenced the work of Albert Einstein Input: A: Isaac Newton, Michael Faraday, and James Clerk Maxwell	SSRT: 1 ZInf: 1 TracIn: 2 LoGra: 835

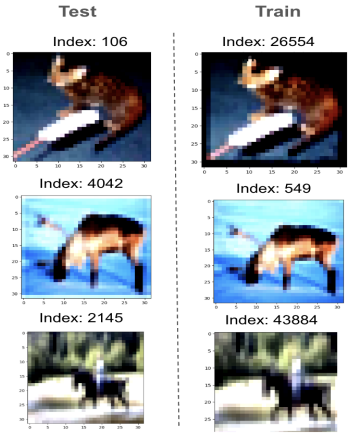

Figure 15: Top-1 Train-Test influence pairs for the CIFAR-10 dataset based on the SSRT, Zeroth and First order approximation methods.

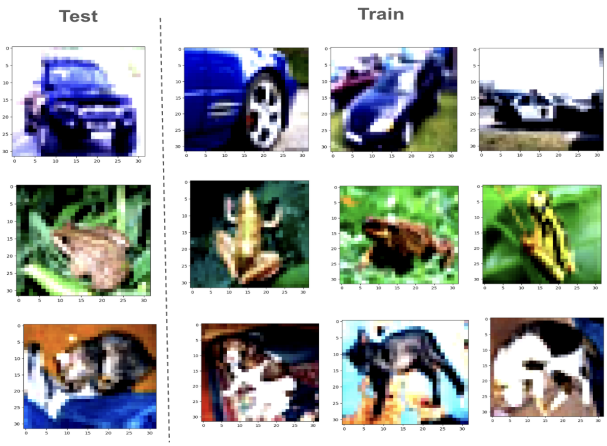

Figure 16: Top-3 train-test influence pairs for the CIFAR10 dataset based on the zeroth-order finite differences gradient approximation.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Section 3 formally describes our method and the theoretical justifications behind it. It explains the relationship to previous work and how a zeroth-order approximation is derived. Subsequent sections show empirically high correlation with the SubSample and Retrain (SSRT) ground truth method for self-influence. Additionally it shows the effectiveness of proposed method for identifying train-test influence and applications in data selection tasks.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We describe the limitations of our work and future directions in Section 5.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not have Theorems and Lemmas. All other theoretical justification and analysis are provided in the paper.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: All required experimental setup and reproducibility requirements are available in the Section 4.1 and appendix.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The code is released on github. The links are provided in the abstract.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.

- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We specify all the training and test details in Section 4.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We observed very small p-values across all SpearmanR correlations tests which validates statistical significance of our experiments. Apart from that we observed insignificant change in the estimates when we considered various combinations of subsets for the SubSample and ReTrain experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

Justification: The requirements about the GPUs, memory and runtime are described in Section 4.1.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The paper conforms with the NeurIPS Code of Ethics in every respect.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Our work can help with data quality controls, auditing and data cleansing for any application and a wide-range of models. Section A.1 presents broader impact in more detail.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This paper does not impose such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We describe the licenses for existing assets in Section A.2 in the Appendix.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.

- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowd-sourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowd-sourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs were not used for ideation or for the development of the method proposed in this paper.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.