

ZOIF: Zeroth-Order Influence Function Estimation for LLM Data Selection Using Only Forward Passes

Lisong He^{*} Yunxuan Jiang[†] Haishan Ye[‡] Xiangyu Chang[§]

School of Management, Xi'an Jiaotong University, Xi'an, China

Abstract

Influence functions enable principled data selection for large language models (e.g., identifying influential training samples for model editing, debugging, or efficient fine-tuning). However, scaling IFs to LLMs is hindered by the intractability of billion-scale Hessian matrices and the prohibitive memory overhead of backpropagation (BP). While many classical IF approximations reduce Hessian complexity, they still rely on BP for gradient computation, requiring massive storage for intermediate activations that impedes practical data selection at scale. To address this, we propose ZOIF, a zeroth-order framework for influence estimation that operates entirely via forward passes, enabling memory-efficient data selection under resource constraints. ZOIF decomposes influence estimation into three BP-free stages: (1) approximating the average proxy gradient via zeroth-order estimation; (2) reformulating and solving the inverse Hessian-vector product as an auxiliary zeroth-order optimization to bypass explicit inversion; and (3) deriving per-sample influence scores via a forward finite difference along the previously computed parameter update direction. Experiments show that ZOIF selects data subsets with downstream utility competitive with—and in some cases exceeding—the evaluated baselines, yet at substantially lower peak memory cost and with comparable overall computation time.

1 Introduction

As large language models (LLMs) continue to demonstrate remarkable performance across a wide range of tasks, the need for efficient, scalable, and principled data selection—such as for fine-tuning, debugging, and model editing—has become increasingly urgent. Influence functions (IFs) originated in robust statistical diagnostics (Hampel,

1974) and were later adapted to modern machine learning (Koh and Liang, 2017). They provide a principled framework for quantifying how individual training samples influence a model’s predictions on test points, thereby offering a theoretically grounded foundation for data attribution and selection (Grosse et al., 2023; Choe et al., 2025; Pan et al., 2024; Xia et al., 2024; Liu et al., 2021).

Formally, IFs quantify the effect of an infinitesimal upweighting of a training sample on a target loss by leveraging a local second-order approximation of the empirical risk. Given a pretrained model with parameters θ_0 , we consider the following local Hessian-preconditioned influence score, which coincides with the classical influence function when θ_0 is a stationary solution of the reference objective:

$$\mathcal{I}(z_i, z'_j) = -\nabla_{\theta} \ell(\theta_0, z_i)^\top \mathbf{H}_{\theta_0}^{-1} \nabla_{\theta} \ell(\theta_0, z'_j), \quad (1)$$

where ℓ is the loss function and $\mathbf{H}_{\theta_0}$ denotes the empirical Hessian of the training objective evaluated at θ_0 . This formulation reveals that influence estimation fundamentally requires solving an inverse Hessian–vector product (IHVP), $\mathbf{H}_{\theta_0}^{-1} \nabla_{\theta} \ell(\theta_0, z'_j)$, which constitutes the primary computational and memory bottleneck when scaling influence functions to modern LLMs. To mitigate this challenge, recent studies have proposed empirical-Fisher or structured-curvature approximations for influence computation (Kwon et al., 2024; Grosse et al., 2023), low-dimensional parameter or curvature subspaces (Schioppa et al., 2022), and random or structured projections of gradient features (Park et al., 2023; Choe et al., 2025). Related second-order approximations have also been explored in model unlearning (Jia et al., 2024).

However, most scalable approximations of classical IFs still rely on first-order (FO) gradients, namely $\nabla_{\theta} \ell(\theta_0, z_i)$ and $\nabla_{\theta} \ell(\theta_0, z'_j)$, which necessitate backpropagation (BP) through the full

^{*}helisong@eurasia.edu (Lisong He)

[†]fengjianliu@stu.xjtu.edu.cn (Yunxuan Jiang)

[‡]yehaishan@xjtu.edu.cn (Haishan Ye)

[§]xiangyuchang@gmail.com (Xiangyu Chang; corresponding author)

computational graph. This dependence incurs prohibitive memory overhead; for instance, computing gradients via BP for OPT-13B requires more than $12\times$ the memory footprint of a single forward pass (Zhang et al., 2024). Under limited GPU memory budgets (common in real-world deployment and edge scenarios), such memory demands severely hinder the practical applicability of influence functions for large-scale data selection.

These limitations motivate the central research question of this work: *How can we estimate IFs for LLMs in a memory-efficient manner, without relying on BP or explicit second-order information, while retaining their effectiveness for data selection?*

To address this question, we propose Z0IF (Zeroth-Order Influence Function Estimation), a novel framework for influence estimation that operates exclusively through forward passes of the model. The key idea underlying Z0IF is to systematically replace all gradient and Hessian-dependent components in classical influence functions with ZO estimators, thereby eliminating the need for BP and the storage of intermediate activations.

At a high level, Z0IF decomposes influence estimation into three memory-efficient stages. First, it approximates the average loss gradient over a small validation subset with ZO evaluations, reducing activation memory while trading it for additional forward passes. Second, Z0IF casts the IHVP as the solution of an auxiliary optimization problem and solves it with forward-only ZO updates. Finally, for each candidate training sample, Z0IF estimates its influence score via a forward finite difference along the previously computed parameter update direction. As designed above, the entire Z0IF pipeline enables scalable, practical influence-based data selection for LLMs under strict memory constraints. The framework of Z0IF is illustrated in Figure 1.

The main contributions of this work can be summarized as follows:

- We introduce Z0IF, a novel ZO framework for IF estimation that operates exclusively via forward passes. By systematically integrating ZO techniques, Z0IF bypasses BP and reduces the activation-memory burden of influence estimation for large-scale models.
- We propose a principled approach to compute the IHVP by reformulating it as an auxiliary ZO optimization problem. This effectively

eliminates the need for computationally expensive high-dimensional matrix inversions. Furthermore, our framework is designed to be highly compatible with various existing influence function approximation techniques, offering significant flexibility for practical deployment across different tasks and resource constraints.

- Experiments show that Z0IF selects data subsets with downstream utility competitive with—and in some cases exceeding—the evaluated baselines, while substantially lowering peak memory usage and maintaining comparable overall computation time.

2 Related Work

During large-scale pretraining and fine-tuning, efficiently identifying high-value samples from massive data pools has become a critical challenge for improving model training efficiency and generalization capability. Existing data selection methods can be broadly categorized into three types: (1) heuristic approaches based on statistical or model-based metrics (Yin and Rush, 2025; Antonello et al., 2021; Marion et al., 2023; Li et al., 2024); (2) similarity-based methods leveraging semantic representations (Ni et al., 2022; Lee et al., 2024; Iverson et al., 2025); and (3) influence-based methods, including gradient-based and forward-only estimators (Pruthi et al., 2020; Wang et al., 2024; Nikdan et al., 2025; Ko et al., 2024; Kokhlikyan et al., 2025; Deng et al., 2026). This paper focuses primarily on the third category and integrates ZO techniques to alleviate the memory bottleneck commonly encountered when applying IFs in large-scale models.

2.1 Approximate Computation of IFs

Scaling IFs to LLMs is primarily limited by Hessian computation, motivating several approximation strategies. One prominent line of research simplifies or discards second-order information to reduce complexity. For instance, TracIn (Pruthi et al., 2020) approximates influence values using inner products of first-order gradients and omits the Hessian term. Influence Distillation retains second-order information when deriving sample weights and introduces a landmark-based approximation for scalable data selection (Nikdan et al., 2025). DataInf employs a damped, layer-wise empirical-Fisher approximation with an efficient closed-form inverse approximation (Kwon et al.,

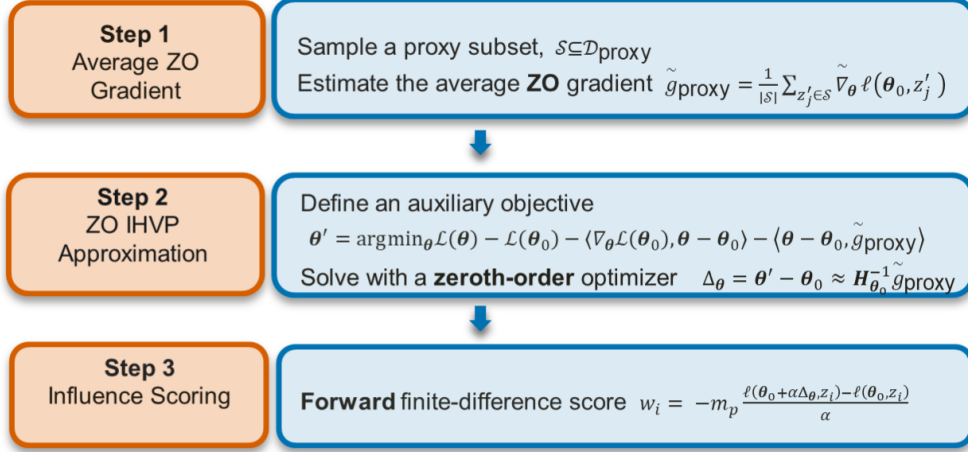


Figure 1: ZOIF framework and information flow across its three forward-only steps.

2024), whereas SOUL uses diagonal second-order information in the related setting of model unlearning (Jia et al., 2024). Another strategy compresses the parameter, curvature, or gradient representation used for influence analysis. Existing methods use low-dimensional parameter or curvature subspaces, structured curvature factorizations, random projections of gradient features, or structured gradient projections (Schioppa et al., 2022; Grosse et al., 2023; Park et al., 2023; Choe et al., 2025). Another line of research focuses on efficiently approximating the IHVP. Classical iterative methods such as conjugate gradient (CG) and LISSA (Agarwal et al., 2017) use repeated Hessian–vector product (HVP) evaluations without explicitly constructing the Hessian. DAUNCE expresses the same local second-order structure as an auxiliary optimization problem (Pan et al., 2025), which motivates the formulation used in Step 2.

2.2 Zeroth-Order Influence Estimation

ZO estimation approximates gradients from forward finite differences along random parameter-space perturbations, avoiding the BP graph and reducing activation memory for resource-constrained LLM fine-tuning (Flaxman et al., 2005; Nesterov and Spokoiny, 2017; Malladi et al., 2023).

Recent forward-only estimators differ in signal and target. Mirrored Influence retains gradients for a small test/proxy set and scores training samples from forward losses (Ko et al., 2024); ZO-Inf uses checkpoint-trajectory loss changes (Kokhlikyan et al., 2025); and For-Value forms a closed-form valuation from final-layer representations

and prediction errors (Deng et al., 2026). ZOIF instead decomposes the classical local Hessian-preconditioned IF into three components that admit ZO estimation: it estimates the proxy gradient with ZO perturbations, obtains the IHVP direction through ZO-based auxiliary optimization, and evaluates each candidate by a forward finite difference along the resulting shared direction. Thus, despite forward-only candidate evaluation, these methods differ in estimand, required information, and computation.

3 Background and Problem Setup

3.1 Data Selection for LLMs

Let an LLM be characterized by a parameter vector $\theta \in \mathbb{R}^d$, where d denotes the total number of trainable parameters. Given a data point $z = (x, y) \in \mathcal{X} \times \mathcal{Y}$, let $\ell(\theta, z)$ denote the loss incurred by the model with parameters θ on z .

Given a candidate data pool $\mathcal{D}_{\text{pool}} = \{z_i = (x_i, y_i)\}_{i=1}^n \subset \mathcal{X} \times \mathcal{Y}$ and an independent validation dataset $\mathcal{D}_{\text{val}} = \{z'_j = (x'_j, y'_j)\}_{j=1}^m \subset \mathcal{X} \times \mathcal{Y}$, we use $\mathcal{D}_{\text{val}}$ to evaluate the model’s generalization performance on the target task. To avoid overfitting to $\mathcal{D}_{\text{val}}$ during the data selection process, it is partitioned into two mutually exclusive subsets: a proxy set $\mathcal{D}_{\text{proxy}} \subset \mathcal{D}_{\text{val}}$, used to guide data selection, with $|\mathcal{D}_{\text{proxy}}| = m_p \ll m$; and a hold-out test set $\mathcal{D}_{\text{test}} = \mathcal{D}_{\text{val}} \setminus \mathcal{D}_{\text{proxy}}$, reserved solely for final unbiased evaluation and kept unseen during the selection phase. The goal of data selection is to choose a subset $\mathcal{D}_{\text{select}} \subseteq \mathcal{D}_{\text{pool}}$ of size n_s such that the model fine-tuned on this subset achieves

optimal performance on $\mathcal{D}_{\text{proxy}}$.

Definition 1. Let θ_0 denote the pretrained initial parameters, and let $\mathcal{A}(\theta_0, \mathcal{D})$ represent the final parameters obtained by applying a fine-tuning algorithm starting from θ_0 using subset $\mathcal{D}$ as training data. The proxy loss is defined as

$$\mathcal{L}_{\text{proxy}}(\theta) = \frac{1}{m_p} \sum_{z'_j \in \mathcal{D}_{\text{proxy}}} \ell(\theta, z'_j). \quad (2)$$

The data selection problem can then be rigorously formulated as the following optimization problem:

$$\mathcal{D}_{\text{select}} \in \arg \min_{\mathcal{D} \subseteq \mathcal{D}_{\text{pool}}, |\mathcal{D}|=n_s} \mathcal{L}_{\text{proxy}}(\mathcal{A}(\theta_0, \mathcal{D})). \quad (3)$$

3.2 Influence Functions

IFs (Koh and Liang, 2017) have emerged as a prominent tool for data selection in recent years, owing to their solid theoretical foundation and strong interpretability. IFs quantify how an infinitesimal upweighting of a training sample z_i (i.e., increasing its weight in the empirical risk) affects the model parameters and the loss on a test point.

The influence score is commonly computed as:

$$\mathcal{I}(z_i, z'_j) = -\nabla_{\theta} \ell(\theta_0, z_i)^\top \mathbf{H}_{\theta_0}^{-1} \nabla_{\theta} \ell(\theta_0, z'_j), \quad (4)$$

where z_i is an arbitrary sample from the data pool, z'_j is an arbitrary sample from $\mathcal{D}_{\text{proxy}}$, $\nabla_{\theta} \ell(\theta_0, z_i) \in \mathbb{R}^d$ is the gradient of the loss with respect to the model parameters, $\mathbf{H}_{\theta_0} = \frac{1}{n} \sum_{i=1}^n \nabla_{\theta}^2 \ell(\theta_0, z_i) \in \mathbb{R}^{d \times d}$ is the empirical Hessian matrix.

The classical upweighting interpretation additionally assumes that θ_0 is a stationary solution of the reference training objective. In practice, one may optionally warm up the model on the reference data so that $\|\nabla \mathcal{L}(\theta_0)\|$ is small. When this warm-up is omitted, as in our experiments to avoid additional training cost, we use the same Hessian-preconditioned gradient similarity as a *local influence proxy* at the given pretrained checkpoint. It recovers the damped classical IF interpretation when the stationarity residual vanishes.

Specifically, if $\mathcal{I}(z_i, z'_j) < 0$, then z_i is a helpful sample whose presence improves model performance on z'_j . By sorting the influence values from smallest to largest, we can prioritize and select the most beneficial training points at the top of the list.

3.3 Zeroth-Order Estimation

Although Hessian approximations avoid explicit construction of a $d \times d$ matrix, conventional IF methods still rely on BP, which stores gradients and intermediate activations and therefore incurs substantial GPU memory overhead for LLMs. We accordingly adopt ZO gradient estimation, which uses only forward evaluations, to reduce this memory cost.

The classical ZO gradient estimator is based on Gaussian smoothing (Nesterov and Spokoiny, 2017). For a target function $\ell(\cdot, z) : \mathbb{R}^d \rightarrow \mathbb{R}$, the forward-difference-based Gaussian-smoothed gradient estimator is defined as:

$$\begin{aligned} \tilde{\nabla}_{\theta} \ell(\theta_0, z) &= \frac{1}{K} \sum_{k=1}^K \frac{\ell(\theta_0 + 2\alpha \mathbf{u}_k, z)}{2\alpha} \mathbf{u}_k \\ &\quad - \frac{1}{K} \sum_{k=1}^K \frac{\ell(\theta_0, z)}{2\alpha} \mathbf{u}_k. \end{aligned} \quad (5)$$

where $\mathbf{u}_k \sim \mathcal{N}(0, I_d)$, $\alpha > 0$ is the smoothing radius, and K is the number of random directions sampled. Its Monte Carlo expectation equals the gradient of the corresponding Gaussian-smoothed objective; it is not generally unbiased for the unsmoothed gradient at finite α . The estimator requires only $K + 1$ forward evaluations, completely avoiding BP and thereby reducing memory overhead.

4 Methodology

4.1 Decomposition of IF Computation

In many practical applications, we are not concerned with the isolated influence of a single training sample on a single proxy sample, but rather with the aggregate influence of each training sample on the entire proxy set. This quantity equals the corresponding row sum of the influence matrix $\mathcal{I} = (\mathcal{I}(z_i, z'_j)) \in \mathbb{R}^{n \times m_p}$. Thus, the core question is *How can we efficiently compute the row sums of the influence matrix without explicitly constructing the full matrix $\mathcal{I}(z_i, z'_j)$?*

We define the total influence of the i -th training sample z_i on the entire proxy set as

$$\begin{aligned} \mathcal{I}_{\text{total}}(z_i) &:= -\sum_{j=1}^{m_p} \nabla_{\theta} \ell(\theta_0, z_i)^\top \mathbf{H}_{\theta_0}^{-1} \nabla_{\theta} \ell(\theta_0, z'_j) \\ &= -m_p \nabla_{\theta} \ell(\theta_0, z_i)^\top \mathbf{H}_{\theta_0}^{-1} \\ &\quad \cdot \left(\frac{1}{m_p} \sum_{j=1}^{m_p} \nabla_{\theta} \ell(\theta_0, z'_j) \right) \\ &= -m_p \nabla_{\theta} \ell(\theta_0, z_i)^\top \mathbf{H}_{\theta_0}^{-1} \mathbf{g}_{\text{proxy}}, \end{aligned}$$

where $\mathbf{g}_{\text{proxy}} = \frac{1}{m_p} \sum_{j=1}^{m_p} \nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0, z'_j)$.

The above formulation provides the following three-step process for the decomposition of total influence computations.

- First, compute the average of gradients $\mathbf{g}_{\text{proxy}}$ across all proxy samples once.
- Second, compute a single IHVP $\mathbf{v} = \mathbf{H}_{\boldsymbol{\theta}_0}^{-1} \mathbf{g}_{\text{proxy}}$.
- Third, for each training sample z_i , obtain its total influence via a simple dot product $-m_p \nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0, z_i)^T \mathbf{v}$.

4.2 ZOIF: Zeroth-Order Influence Estimation

Building on the above decomposition, we propose a fully ZO, three-stage computational framework to efficiently estimate IFs for LLMs under memory-constrained conditions. The entire pipeline operates without explicit access to gradients or Hessian matrices, relying solely on model forward passes.

4.2.1 Step 1: Average ZO Gradient Estimation

Directly computing the full gradient averaged over all m_p proxy samples is computationally prohibitive under memory-constrained conditions. To address this, we proceed as follows: First, we uniformly sample a subset $\mathcal{S} \subseteq \mathcal{D}_{\text{proxy}}$ of size $|\mathcal{S}|$ from the proxy dataset. Then, for each $z'_j \in \mathcal{S}$, we approximate the gradient using ZOGE (Zeroth-Order Gradient Estimation; see the supplementary material), following Malladi et al. (2023)¹, and aggregate the results into an averaged gradient estimate:

$$\tilde{\mathbf{g}}_{\text{proxy}} = \frac{1}{|\mathcal{S}|} \sum_{z'_j \in \mathcal{S}} \tilde{\nabla}_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0, z'_j) \quad (6)$$

where $\tilde{\nabla}_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0, z'_j)$ is denoted in Eq.(5).

The trade-off between computational cost and estimation variance can be controlled by adjusting $|\mathcal{S}|$. In particular, when the proxy dataset size m_p is small, we can set $|\mathcal{S}| = m_p$ to remove proxy-subsampling error. The remaining discrepancy from the unsmoothed average gradient consists of Monte Carlo sketching error and finite-difference/smoothing bias.

¹We adopt the gradient estimator from ZO optimization but replace central differences with forward differences, reducing function evaluations from $2K$ to $K + 1$ for K random directions.

4.2.2 Step 2: Solving the IHVP via ZO Optimization

Following the auxiliary optimization formulation introduced by DAUNCE (Pan et al., 2025), we recast the computation of the IHVP as an optimization problem that can be solved entirely with ZO updates.

Let $\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{n} \sum_{z \in \mathcal{D}_{\text{pool}}} \ell(\boldsymbol{\theta}, z)$, and estimate the reference gradient $\tilde{\mathbf{g}}_0$ using only forward evaluations. Starting from $\boldsymbol{\theta}_0$, we seek a local stationary solution $\boldsymbol{\theta}'$ of the objective

$$\mathcal{J}(\boldsymbol{\theta}) = (\mathcal{L}(\boldsymbol{\theta}) - \mathcal{L}(\boldsymbol{\theta}_0) - \langle \tilde{\mathbf{g}}_0, \boldsymbol{\theta} - \boldsymbol{\theta}_0 \rangle - \langle \boldsymbol{\theta} - \boldsymbol{\theta}_0, \tilde{\mathbf{g}}_{\text{proxy}} \rangle). \quad (7)$$

This optimization problem is derived from the second-order Taylor expansion of the loss function, where

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}) - \mathcal{L}(\boldsymbol{\theta}_0) - \langle \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_0), \boldsymbol{\theta} - \boldsymbol{\theta}_0 \rangle \\ \approx \frac{1}{2} (\boldsymbol{\theta} - \boldsymbol{\theta}_0)^T \mathbf{H}_{\boldsymbol{\theta}_0} (\boldsymbol{\theta} - \boldsymbol{\theta}_0). \end{aligned} \quad (8)$$

After local optimization, the resulting parameter change $\Delta_{\boldsymbol{\theta}} = \boldsymbol{\theta}' - \boldsymbol{\theta}_0$ approximates $\mathbf{H}_{\boldsymbol{\theta}_0}^{-1} \tilde{\mathbf{g}}_{\text{proxy}}$ when the local curvature is positive definite and the stationarity residual is small. Both the reference gradient and the auxiliary-objective updates are estimated by ZOGE, and we project $\Delta_{\boldsymbol{\theta}}$ onto a small Euclidean ball for numerical stability. When additional stabilization is needed, adding $\lambda \|\boldsymbol{\theta} - \boldsymbol{\theta}_0\|_2^2 / 2$ yields the damped direction $(\mathbf{H}_{\boldsymbol{\theta}_0} + \lambda \mathbf{I})^{-1} \tilde{\mathbf{g}}_{\text{proxy}}$.

Both $\tilde{\mathbf{g}}_0$ and the iterative solution of Eq. (7) are computed with ZO estimators (Malladi et al., 2023); consequently, Step 2 does not require BP or access to exact gradients. Algorithm 1 presents the full-objective form for clarity, while the implementation uses stochastic minibatch approximations of the same local objective. The theoretical stationarity residual in Theorem 1 records the combined error of this finite stochastic solve.

Step 2 also admits alternative ZO-compatible approximations of the inverse-Hessian action.²

4.2.3 Step 3: ZO Directional Influence Score Estimation

Following the ZO paradigm, where gradients are approximated using only loss evaluations, we retain the core ZO rationale but adopt a more efficient strategy that bypasses explicit per-sample gradient

²Detailed formulations and score-error extensions for the plug-in Step 2 variants are provided in the supplementary material.

Algorithm 1 ZOIF: Zeroth-Order Influence Function Estimation

Require: $\mathcal{D}_{\text{pool}}$, $\mathcal{D}_{\text{proxy}}$, LLM model initial parameters θ_0 and loss function ℓ , proxy-set size m_p , ZO parameters (K, α) , size $|S|$.

Ensure: Estimated influence scores $\{w_i\}_{i=1}^n$.

Step 1: Average ZO Gradient Estimation

- 1: Randomly select a subset $\mathcal{S} \subseteq \mathcal{D}_{\text{proxy}}$ of size $|S|$
 - 2: Compute average ZO gradient estimate:
 - 3: $\tilde{g}_{\text{proxy}} = \frac{1}{|S|} \sum_{z'_j \in \mathcal{S}} \text{ZOG}(\ell, \theta_0, z'_j, K, \alpha)$
-

Step 2: Solving the IHVP via ZO Optimization

- 4: Compute the ZO reference-gradient estimate $\tilde{g}_0 \leftarrow \text{ZOG}(\mathcal{L}, \theta_0)$
 - 5: Define auxiliary optimization problem as Eq. (7)
 - 6: Solve for θ' using a ZO iterative solver and project the update onto the prescribed norm ball
 - 7: Get $\Delta_\theta \leftarrow \theta' - \theta_0$
-

Step 3: ZO Directional Influence Score Estimation

- 8: **for** each data pool sample z_i **do**
 - 9: Compute: $\ell_0 \leftarrow \ell(\theta_0, z_i)$
 - 10: Compute: $\ell^+ \leftarrow \ell(\theta_0 + \alpha \Delta_\theta, z_i)$
 - 11: Estimate influence score: $w_i \leftarrow -m_p \cdot \frac{\ell^+ - \ell_0}{\alpha}$
 - 12: **end for**
 - 13: **Return** $\{w_i\}_{i=1}^n$
-

reconstruction. Specifically, we directly approximate the directional derivative $\nabla_\theta \ell(\theta_0, z_i)^\top \Delta_\theta$ via a forward finite difference along the previously computed parameter update direction Δ_θ :

$$w_i = -m_p \cdot \frac{\ell(\theta_0 + \alpha \Delta_\theta, z_i) - \ell(\theta_0, z_i)}{\alpha}, \quad (9)$$

which converges to $-m_p \nabla_\theta \ell(\theta_0, z_i)^\top \Delta_\theta$ as $\alpha \rightarrow 0$. This formulation requires two forward passes per sample and avoids the backward pass and per-sample gradient storage required by FOIF.

Based on this, we rank the samples in $\mathcal{D}_{\text{pool}}$ by their influence scores in ascending order and select the top n_s samples with the smallest influence for subsequent fine-tuning.

Remark 1. For full-parameter LLMs with batch size $B = 1$, relative to FOIF, ZOIF saves $A_{\text{activations}} - A_{\text{max}}$ peak memory in Step 1, at least the same amount per gradient evaluation in Step 2, and $d\beta + A_{\text{activations}} - A_{\text{max}}$ in Step 3. Here, $A_{\text{activations}} = L(HS^2 + SD_{\text{ffn}})\beta$ approximates cumulative BP activation memory, and A_{max} is the largest single-layer activation retained by a forward

evaluation; full notation and derivations are given in the supplementary material.

4.3 Theoretical Properties

This section provides the theoretical analysis for the proposed ZOIF. We establish upper bounds on the deviation between the influence scores estimated by ZOIF and FOIF under two settings. First, we consider a locally invertible Hessian in the full parameter space. We then relax this condition to a low-rank Hessian structure that better reflects the local geometry of LLMs. All proofs are provided in the supplementary material.

4.3.1 Deviation Between ZOIF and FOIF

Assumption 1. There exists $M > 0$ such that the per-sample Hessian norm is at most M along every ZO perturbation segment used in Steps 1–2 and every Step 3 segment considered below.

Assumption 2. There exists $G > 0$ such that $\|\nabla \ell(\theta_0, z)\|_2 \leq G$ for every proxy, reference, and scored sample z .

Assumption 3. Along the local Step 2 segment $\{\theta_0 + t\Delta_\theta : t \in [0, 1]\}$, $\nabla^2 \mathcal{L}(\theta) \succeq \mu \mathbf{I}$ for some $\mu > 0$. In particular, $\|\mathbf{H}_{\theta_0}^{-1}\|_2 \leq 1/\mu$, and the integrated Hessian

$$\overline{\mathbf{H}}(\Delta_\theta) = \int_0^1 \nabla^2 \mathcal{L}(\theta_0 + t\Delta_\theta) dt$$

also has inverse norm at most $1/\mu$.

Assumption 4. The per-sample Hessians on the above ZO and scoring segments, and the Hessian of $\mathcal{L}$ on the local Step 2 segment, are L_H -Lipschitz.

Let τ bound the first-order residual of the finite Step 2 output:

$$\|\nabla \mathcal{L}(\theta_0 + \Delta_\theta) - \tilde{g}_0 - \tilde{g}_{\text{proxy}}\|_2 \leq \tau.$$

The case $\tau = 0$ is a local stationary solution; no global minimizer is required.

Theorem 1. Under Assumptions 1–4, suppose the Step 2 norm projection is inactive. For each scored sample z_i , let $\mathcal{I}_{\text{FO}}(z_i)$ denote its FOIF score. Conditional on the sampled subsets, for any $0 < \delta_0 < 1$, with probability at least $1 - \delta_0$ over the Gaussian directions, the outputs w_i of Algorithm 1 satisfy, si-

multaneously for all scored samples z_i ,

$$\begin{aligned} & |\mathcal{I}_{\text{FO}}(z_i) - w_i| \\ & \leq \frac{m_p G E_\tau}{\mu} + \frac{m_p L_H G^2 (G + E_\tau)}{2\mu^3} \\ & \quad + \frac{m_p M \alpha}{2} \left(\frac{G + E_\tau}{\mu} \right)^2 \\ & \quad + \frac{m_p L_H \alpha^2}{6} \left(\frac{G + E_\tau}{\mu} \right)^3. \end{aligned}$$

where

$$\begin{aligned} E_\tau &= \varepsilon_{\text{sub}} + \varepsilon_{\text{ref}} + \tau \\ &+ \mathcal{E}_d(q_p, K, \alpha, \delta_0/2) + \mathcal{E}_d(q_0, K, \alpha, \delta_0/2), \end{aligned}$$

and

$$\begin{aligned} \mathcal{E}_d(q, K, \alpha, \delta) &= G \sqrt{\frac{d+1}{qK\delta}} \\ &+ \frac{\alpha M}{qK} \sum_{j=1}^q \sum_{k=1}^K \|\mathbf{u}_{jk}\|_2^3 \\ &+ \frac{2L_H \alpha^2}{3qK} \sum_{j=1}^q \sum_{k=1}^K \|\mathbf{u}_{jk}\|_2^4. \end{aligned}$$

Here, q_p , q_0 , ε_{sub} , and ε_{ref} are defined in the supplementary material. Thus, increasing K reduces the Gaussian sketching error as $O(K^{-1/2})$, while decreasing α reduces the finite-radius and Step 3 errors, which scale as $O(\alpha) + O(\alpha^2)$.

As a ranking consequence, if the right-hand side of Theorem 1 is uniformly at most $\varepsilon_{\text{score}}$, the ascending Top- n_s set is preserved when the FOIF boundary gap exceeds $2\varepsilon_{\text{score}}$.

4.3.2 Deviation under Low-Rank Hessian Structure

While Theorem 1 assumes positive curvature in the full parameter space, pre-trained LLMs can exhibit highly anisotropic local curvature, with most Hessian eigenvalues concentrated near zero and only a small number of dominant directions (Malladi et al., 2023). The full-space condition can therefore be restrictive. To address this, we replace it with a low-rank non-degeneracy condition and analyze the corresponding projected FOIF target.

Assumption 5. There is an orthonormal basis $\mathbf{U}_r \in \mathbb{R}^{d \times r}$ such that Step 2 and all ZO perturbations are restricted to $\text{span}(\mathbf{U}_r)$. The projected reference and integrated Hessians satisfy the local curvature condition in Assumption 3. Define

$$\mathbf{H}_{\theta_0}^\dagger = \mathbf{U}_r (\mathbf{U}_r^\top \mathbf{H}_{\theta_0} \mathbf{U}_r)^{-1} \mathbf{U}_r^\top.$$

Based on this assumption, the following theorem characterizes the estimation error in the low-rank setting.

Theorem 2. Under Assumptions 1–4 and 5, suppose the Step 2 norm projection is inactive. Let $\mathcal{I}_{\text{FO}}^{(r)}(z_i)$ denote the projected FOIF score obtained using $\mathbf{H}_{\theta_0}^\dagger$. Conditional on the sampled subsets, for any $0 < \delta_0 < 1$, with probability at least $1 - \delta_0$ over the Gaussian directions, simultaneously for all scored samples z_i ,

$$\begin{aligned} |\mathcal{I}_{\text{FO}}^{(r)}(z_i) - w_i| &\leq \frac{m_p G E_{r,\tau}}{\mu} \\ &+ \frac{m_p L_H G^2 (G + E_{r,\tau})}{2\mu^3} \\ &+ \frac{m_p M \alpha}{2} \left(\frac{G + E_{r,\tau}}{\mu} \right)^2 \\ &+ \frac{m_p L_H \alpha^2}{6} \left(\frac{G + E_{r,\tau}}{\mu} \right)^3, \end{aligned}$$

where

$$\begin{aligned} E_{r,\tau} &= \varepsilon_{\text{sub}} + \varepsilon_{\text{ref}} + \tau \\ &+ \mathcal{E}_r(q_p, K, \alpha, \delta_0/2) \\ &+ \mathcal{E}_r(q_0, K, \alpha, \delta_0/2), \end{aligned}$$

and $\mathcal{E}_r$ is the subspace counterpart of $\mathcal{E}_d$, evaluated with rank r and perturbations restricted to $\text{span}(\mathbf{U}_r)$.

Thus, under the explicit rank- r restriction, the sketching error depends on the effective dimension r rather than the ambient dimension d , and the deviation from the projected FOIF target remains controlled when $E_{r,\tau}$ and the finite-difference terms are small.

5 Experiment Study

5.1 Experimental Design

5.1.1 Data Pool and Validation Data

Following the experimental setup of Ivison et al. (2023, 2025), we randomly sample a subset of 200,000 examples from the TULU2 dataset to form the candidate data pool. From this pool, we further select 10,000 samples for subsequent model fine-tuning. To ensure fairness and comparability, we evaluate our approach on three widely used benchmark datasets: MMLU (Hendrycks et al., 2021), BBH (Suzgun et al., 2022), and SQuAD (Rajpurkar et al., 2016). Detailed evaluation protocols are provided in the supplementary material.

Peak GPU memory (GB)											
LLM	FOIF					ZOIF					Mode
	S1	dia	hvp	opt	S3	S1	dia	hvp	opt	S3	
pythia-70m	0.68	0.94	0.86	2.70	0.92	0.63	0.90	0.63	1.72	0.51	Full
Llama-3.2-1B	3.91	3.94	3.96	10.67	3.91	2.96	2.97	2.97	3.00	2.95	LoRA
Llama-2-7B	15.74	15.56	15.62	34.39	15.46	13.08	13.15	13.15	13.27	13.02	LoRA
Qwen3-0.6B	3.09	3.11	3.13	15.77	3.10	1.92	1.94	1.94	1.98	1.91	LoRA
Qwen3-1.7B	5.68	5.73	5.74	21.46	5.69	4.03	4.06	4.06	4.11	4.01	LoRA
Runtime (hours)											
LLM	FOIF					ZOIF					Mode
	S1	dia	hvp	opt	S3	S1	dia	hvp	opt	S3	
pythia-70m	0.017	0.021	0.040	0.033	37.60	0.27	0.33	0.58	0.60	22.50	Full
Llama-3.2-1B	0.058	0.083	0.175	0.050	34.00	0.80	0.92	2.28	0.80	30.00	LoRA
Llama-2-7B	0.100	0.150	0.320	0.050	43.00	2.00	2.50	5.00	2.00	40.00	LoRA
Qwen3-0.6B	0.100	0.150	0.310	0.050	40.00	1.50	1.70	3.50	1.00	31.75	LoRA
Qwen3-1.7B	0.100	0.150	0.320	0.050	40.30	1.50	1.80	3.50	1.60	32.00	LoRA

Table 1: Step-wise peak GPU memory and runtime for FOIF and ZOIF. S1 and S3 denote Steps 1 and 3; the dia, hvp, and opt columns report the corresponding Step 2 variants.

Because the evaluation spans multiple tasks, we consider two data selection strategies. **Strategy 1** aggregates influence scores across tasks to produce a single ranking for joint fine-tuning and evaluation. **Strategy 2** ranks and selects data separately for each task, followed by task-specific fine-tuning and evaluation.

5.1.2 LLMs

We use full-parameter influence computation and fine-tuning for pythia-70m (Biderman et al., 2023). For Llama-3.2-1B (Dubey et al., 2024), Llama-2-7B (Touvron et al., 2023), Qwen3-0.6B, and Qwen3-1.7B (Team, 2025), we perform both influence estimation and fine-tuning in the LoRA parameter subspace. Full parameter settings are provided in the supplementary material.

5.1.3 Data Selection Methods

Our Methods We evaluate the optimization-based method ZOIF_opt together with three plug-in variants: ZOIF_hvp uses a DataInf-style damped block-wise empirical-Fisher inverse approximation (Kwon et al., 2024); ZOIF_dia uses a ZO-compatible diagonal preconditioner; and ZOIF_ide replaces the curvature operator with the identity. All gradient-dependent quantities in these variants are supplied by ZO estimates. Their unified mathematical form and variant-specific residuals are given in the supplementary material.

For ZO gradient estimation via perturbation, we set $K = 30$ and $\alpha = 1e-3$.³

³Sensitivity analyses of K , α , and the data selection ratio,

Baseline Methods We compare against: (1) FOIF_opt, FOIF_hvp, FOIF_dia, and FOIF_ide, the first-order counterparts of the corresponding ZOIF variants; (2) LESS (Xia et al., 2024) and TRAK (Park et al., 2023), gradient-based attribution methods using projection or sketching; (3) EMB, based on embedding similarity (Ni et al., 2022; Lee et al., 2024); (4) Top_PPL and Mid_PPL, based on perplexity (Yin and Rush, 2025); (5) Random, which uniformly selects 10,000 examples; and (6) Orig, the original model without fine-tuning. All experiments are conducted in the same environment, with results averaged over three independent runs.

5.2 Experimental Results

5.2.1 Memory and Runtime Efficiency

Memory Efficiency Table 1 reports peak GPU memory across all three steps. ZOIF uses less memory than its FOIF counterpart in every evaluated setting, with the largest reductions appearing in the Step 2 optimization variant.⁴

Time Efficiency Table 1 also provides the step-wise runtime. ZO perturbations increase the cost of Steps 1 and 2, whereas the two-forward-pass score

together with the ranking correlation between ZOIF and FOIF, are provided in the supplementary material.

⁴We further compare ZOIF against FOIF augmented with standard memory optimization techniques (Gradient Checkpointing and Activation Offloading) in Step 1 on Llama-3.2-1B. Results show that even when FOIF employs both optimizations simultaneously in this step, ZOIF still retains a clear memory advantage; see the supplementary material for details.

in Step 3 is faster than FO scoring. Because Step 3 dominates the end-to-end workload, the overall runtime remains comparable across the evaluated models.

5.2.2 Data Selection Performance

We report a representative result below; complete evaluations are provided in the supplementary material.

Results on Qwen Results for Qwen3-0.6B are presented in Table 2. On this model, ZOIF_dia achieves the highest average score (51.97), compared with LESS (51.68), TRAK (50.61), and the evaluated FOIF variants. Task-dependent rankings among the curvature variants motivate task-specific operator selection. Additional model results, sensitivity analyses, and stability tests are reported in the supplement.

Method	MMLU	BBH	SQuAD	Avg.
ZOIF_dia	44.09	34.67	77.17	51.97 $\pm$ 0.32
ZOIF_hvp	43.89	34.38	74.70	50.99 $\pm$ 0.96
ZOIF_ide	44.50	34.32	76.30	51.71 $\pm$ 0.44
ZOIF_opt	44.41	34.81	73.42	50.88 $\pm$ 0.76
FOIF_dia	43.95	34.19	72.38	50.17 $\pm$ 1.02
FOIF_hvp	43.89	33.77	75.21	50.96 $\pm$ 0.65
FOIF_ide	44.16	32.23	67.20	47.86 $\pm$ 1.77
FOIF_opt	45.22	34.08	72.61	50.64 $\pm$ 1.12
LESS	43.22	33.37	78.45	51.68 $\pm$ 0.23
TRAK	42.97	32.83	76.02	50.61 $\pm$ 0.84
EMB	45.30	33.72	70.91	49.98 $\pm$ 0.52
Top_PPL	45.75	34.00	71.52	50.43 $\pm$ 0.27
Mid_PPL	45.82	33.83	72.31	50.66 $\pm$ 0.24
Random	45.27	34.33	69.51	49.70 $\pm$ 0.37
Orig	47.09	30.54	48.45	42.03

Table 2: Held-out Qwen3-0.6B performance (Strategy 1). Avg. is the macro-average of MMLU/BBH accuracy and SQuAD F1. The Avg. entry is reported as mean $\pm$ standard deviation over three independent runs.

6 Conclusion

ZOIF is a forward-only framework for influence-based data selection under limited GPU memory. Its analysis separates numerical, curvature-operator, and directional finite-difference errors and extends to low-rank subspaces. Across full-parameter and LoRA experiments on five language models, it substantially reduces peak memory while retaining competitive subset utility, offering a practical time–memory trade-off.

References

- Naman Agarwal, Brian Bullins, and Elad Hazan. 2017. Second-order stochastic optimization for machine learning in linear time. *The Journal of Machine Learning Research*, 18(1):4148–4187.
- Richard Antonello, Nicole Beckage, Javier Turek, and Alexander Huth. 2021. Selecting informative contexts improves language model fine-tuning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1072–1085.
- Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O’Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.
- Sang Choe, Hwijee Ahn, Juhan Bae, Kewen Zhao, Youngseog Chung, Adithya Pratapa, Willie Neiswanger, Emma Strubell, Teruko Mitamura, Jeff G. Schneider, Eduard Hovy, Roger Baker Grosse, and Eric P. Xing. 2025. What is Your Data Worth to GPT? LLM-Scale Data Valuation with Influence Functions. In *NeurIPS*, volume 38, pages 145944–145985.
- Wenlong Deng, Qi Zeng, Jiaming Zhang, Minghui Chen, Zixin Ding, Christos Thrampoulidis, Boying Gong, and Xiaoxiao Li. 2026. [For-value: Efficient forward-only data valuation for finetuning LLMs and VLMs](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14581–14600. Association for Computational Linguistics.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.
- Abraham D. Flaxman, Adam Tauman Kalai, and H. Brendan McMahan. 2005. Online convex optimization in the bandit setting: Gradient descent without a gradient. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 385–394.
- Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, et al. 2023. Studying large language model generalization with influence functions. *arXiv preprint arXiv:2308.03296*.
- Frank R Hampel. 1974. The influence curve and its role in robust estimation. *Journal of the American statistical association*, 69(346):383–393.

- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. 2023. [Camels in a changing climate: Enhancing lm adaptation with tulu 2](#). *Preprint*, arXiv:2311.10702.
- Hamish Ivison, Muru Zhang, Faeze Brahman, Pang Wei Koh, and Pradeep Dasigi. 2025. Large-scale data selection for instruction tuning. *arXiv preprint arXiv:2503.01807*.
- Jinghan Jia, Yihua Zhang, Yimeng Zhang, Jiancheng Liu, Bharat Runwal, James Diffenderfer, Bhavya Kaikhura, and Sijia Liu. 2024. [SOUL: Unlocking the power of second-order optimization for LLM unlearning](#). *Preprint*, arXiv:2404.18239.
- Myeongseob Ko, Feiyang Kang, Weiyan Shi, Ming Jin, Zhou Yu, and Ruoxi Jia. 2024. [The mirrored influence hypothesis: Efficient data influence estimation by harnessing forward passes](#). In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26286–26295.
- Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 1885–1894. PMLR.
- Narine Kokhlikyan, Kamalika Chaudhuri, and Saeed Mahlouljifar. 2025. [Z0-Inf: Zeroth order approximation for data influence](#). *Preprint*, arXiv:2510.11832.
- Yongchan Kwon, Eric Wu, Kevin Wu, and James Zou. 2024. [DataInf: Efficiently estimating data influence in lora-tuned llms and diffusion models](#). In *The Twelfth International Conference on Learning Representations*.
- Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoneyi, Bryan Catanzaro, and Wei Ping. 2024. NV-Embed: Improved techniques for training llms as generalist embedding models. *arXiv preprint arXiv:2405.17428*.
- Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 2024. From quantity to quality: Boosting llm performance with self-guided data selection for instruction tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 7602–7635.
- Zhuoming Liu, Hao Ding, Huaping Zhong, Weijia Li, Jifeng Dai, and Conghui He. 2021. Influence selection for active learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9274–9283.
- Ilya Loshchilov and Frank Hutter. 2019. [Decoupled weight decay regularization](#). *Preprint*, arXiv:1711.05101.
- Sadhika Malladi, Tianyu Gao, Eshaan Nichani, Alex Damian, Jason D. Lee, Danqi Chen, and Sanjeev Arora. 2023. [Fine-tuning language models with just forward passes](#). In *NeurIPS*, volume 36, pages 53038–53075.
- Max Marion, Ahmet Üstün, Luiza Pozzobon, Alex Wang, Marzieh Fadaee, and Sara Hooker. 2023. When less is more: Investigating data pruning for pre-training llms at scale. *arXiv preprint arXiv:2309.04564*.
- Yu. Nesterov and V. Spokoiny. 2017. Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics*, 17:527–566.
- Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernandez Abrego, Ji Ma, Vincent Zhao, Yi Luan, Keith Hall, Ming-Wei Chang, and Yinfei Yang. 2022. Large dual encoders are generalizable retrievers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9844–9855.
- Mahdi Nikdan, Vincent Cohen-Addad, Dan Alistarh, and Vahab Mirrokni. 2025. [Efficient data selection at scale via influence distillation](#). In *Advances in Neural Information Processing Systems*, volume 38.
- Xingyuan Pan, Luyang Huang, Liyan Kang, Zhicheng Liu, Yu Lu, and Shanbo Cheng. 2024. G-dig: Towards gradient-based diverse and high-quality instruction data selection for machine translation. *arXiv preprint arXiv:2405.12915*.
- Xingyuan Pan, Chenlu Ye, Joseph Melkonian, Jiaqi W Ma, and Tong Zhang. 2025. DAUNCE: Data attribution through uncertainty estimation. *arXiv preprint arXiv:2505.23223*.
- Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. 2023. [TRAK: Attributing model behavior at scale](#). In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 27074–27113. PMLR.
- Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. 2020. Estimating training data influence by tracing gradient descent. In *Advances in Neural Information Processing Systems*.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. [SQuAD: 100,000+ questions for machine comprehension of text](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.
- Andrea Schioppa, Polina Zablotskaia, David Vilar, and Artem Sokolov. 2022. Scaling up influence functions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8179–8186.

- Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, , and Jason Wei. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.
- Qwen Team. 2025. [Qwen3 technical report](#). *Preprint*, arXiv:2505.09388.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.
- Henrique Schechter Vera, Sahil Dua, Biao Zhang, Daniel Salz, Ryan Mullins, Sindhu Raghuram Panayam, Sara Smoot, Iftekhar Naim, Joe Zou, Feiyang Chen, et al. 2025. [EmbeddingGemma: Powerful and lightweight text representations](#). *Preprint*, arXiv:2509.20354.
- Jiachen T. Wang, Tong Wu, Dawn Song, Prateek Mittal, and Ruoxi Jia. 2024. [GREATS: Online selection of high-quality data for LLM training in every iteration](#). In *Advances in Neural Information Processing Systems*, volume 37.
- Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. 2024. LESS: Selecting influential data for targeted instruction tuning. In *ICML*, pages 54104–54132.
- Junjie Oscar Yin and Alexander M. Rush. 2025. Compute-constrained data selection. In *ICLR*.
- Ci Zhang, Chence Yang, Qitao Tan, Jun Liu, Ao Li, Yanzhi Wang, Jin Lu, Jinhui Wang, and Geng Yuan. 2025. Towards memory-efficient and sustainable machine unlearning on edge using zeroth-order optimizer. In *Proceedings of the Great Lakes Symposium on VLSI 2025*, pages 227–232.
- Yihua Zhang, Pingzhi Li, Junyuan Hong, Jiaxiang Li, Yimeng Zhang, Wenqing Zheng, Pin-Yu Chen, Jason D Lee, Wotao Yin, Mingyi Hong, et al. 2024. Revisiting zeroth-order optimization for memory-efficient llm fine-tuning: A benchmark. *arXiv preprint arXiv:2402.11592*.

A ZOGE Algorithm

The ZOGE procedure follows Malladi et al. (2023) and is summarized in Algorithm 2.

B Memory and Computational Complexity

B.1 Memory Analysis

To compare the memory overhead of ZOIF and FOIF, we use the following variables for model architecture and computation:

- d : total number of model parameters (e.g., 7B, 13B);
- S : input sequence length (in tokens);
- L : number of Transformer layers;
- H : number of attention heads;
- D_{model} : hidden dimension;
- $D_{\text{ffn}} \approx 4D_{\text{model}}$: intermediate dimension of the feed-forward network (FFN);
- B : batch size;
- $\beta \in \{2, 4\}$: bytes per parameter (2 for bfloat16/float16, 4 for float32).

B.1.1 Step 1

In Step 1, we estimate the average gradient over the subset drawn from the proxy dataset. For full-parameter computation with $B = 1$, both methods require loading the model parameters (M_{params}), computing the gradient for a single sample (G_{grads}), and accumulating the total gradient in a gradient pool ($G_{\text{gradspool}}$).

For FO methods, the peak memory consumption is

$$\text{Mem}_1 = M_{\text{params}} + G_{\text{grads}} + A_{\text{activations}} + G_{\text{gradspool}},$$

where $M_{\text{params}} = G_{\text{grads}} = G_{\text{gradspool}} = d\beta$. The term $A_{\text{activations}}$ accounts for all intermediate tensors required to support backpropagation, including both the computational graph metadata and, dominantly, the cached activations from each layer. In Transformers, these activations are primarily the self-attention scores ($O(HS^2)$) and FFN intermediate outputs ($O(SD_{\text{ffn}})$), leading to

$$A_{\text{activations}} \approx L(HS^2 + SD_{\text{ffn}})\beta.$$

Algorithm 2 ZOGE: Zeroth-Order Gradient Estimation

Require: LLM model with parameters $\theta_0 = \{\theta_j\}_{j=1}^d$, loss function ℓ , input sample z , number of perturbations K , perturbation radius $\alpha > 0$.

Ensure: Estimated gradient $\tilde{\nabla}_{\theta}\ell(\theta_0, z)$.

```

1: Initialize estimate:  $\tilde{\nabla}_{\theta}\ell(\theta_0, z) \leftarrow \mathbf{0}$  (same shape as  $\theta_0$ )
2: Initialize valid counter:  $C_{\text{valid}} \leftarrow 0$ 
3: coef, seeds  $\leftarrow []$ 
4: Compute baseline loss:  $\ell_0 \leftarrow \ell(\theta_0, z)$ 
5: if  $\ell_0$  is NaN or Inf then
6:   return  $\mathbf{0}$  {or skip this sample in outer loop}
7: end if
8: for  $i = 1, 2, \dots, K$  do
9:   Sample a random seed  $s_i$ 
10:  Set random number generator seed to  $s_i$ 
11:  for each parameter  $\theta_j \in \theta_0$  do
12:    Sample scalar  $u_j \sim \mathcal{N}(0, 1)$ 
13:    Perturb in-place:  $\theta_j \leftarrow \theta_j + 2\alpha \cdot u_j$ 
14:  end for
15:  Compute  $\ell^+ \leftarrow \ell(\theta_0, z)$ 
16:  Reset random number generator seed to  $s_i$ 
17:  for each parameter  $\theta_j \in \theta_0$  do
18:    Sample scalar  $u_j \sim \mathcal{N}(0, 1)$ 
19:    Restore:  $\theta_j \leftarrow \theta_j - 2\alpha \cdot u_j$ 
20:  end for
21:  if  $\ell^+$  is NaN or Inf then
22:    continue
23:  end if
24:  Compute coefficient:  $c_i \leftarrow \frac{\ell^+ - \ell_0}{2\alpha}$ 
25:  Append  $s_i$  to seeds
26:  Append  $c_i$  to coef
27:   $C_{\text{valid}} \leftarrow C_{\text{valid}} + 1$ 
28: end for
29: for  $i = 1, 2, \dots, C_{\text{valid}}$  do
30:  Reset random number generator seed to seeds[ $i$ ]
31:   $c_i = \text{coef}[i]$ 
32:  for each parameter  $\theta_j \in \theta_0$  do
33:    Sample scalar  $u_j \sim \mathcal{N}(0, 1)$ 
34:     $(\tilde{\nabla}_{\theta}\ell(\theta_0, z))_j \leftarrow (\tilde{\nabla}_{\theta}\ell(\theta_0, z))_j + c_i \cdot u_j$ 
35:  end for
36: end for
37: if  $C_{\text{valid}} > 0$  then
38:    $\tilde{\nabla}_{\theta}\ell(\theta_0, z) \leftarrow \frac{1}{C_{\text{valid}}} \tilde{\nabla}_{\theta}\ell(\theta_0, z)$ 
39: else
40:    $\tilde{\nabla}_{\theta}\ell(\theta_0, z) \leftarrow \mathbf{0}$ 
41: end if
42: Return  $\tilde{\nabla}_{\theta}\ell(\theta_0, z)$ 

```

Thus, the total memory usage is

$$\text{Mem}_1(S) = 3d\beta + L(HS^2 + SD_{\text{ffn}})\beta.$$

In contrast, ZO methods estimate gradients by perturbing model parameters and measuring the resulting change in loss. Following recent works (e.g., [Malladi et al. \(2023\)](#); [Zhang et al. \(2025\)](#)) and as illustrated in Algorithm 2, advanced ZO strategies avoid storing full perturbation vectors. Instead, only random seeds are retained; during parameter updates, a scalar Gaussian noise $z \sim \mathcal{N}(0, 1)$ is independently sampled for each parameter θ_i , and the gradient projection is applied directly. The final gradient estimate is reconstructed on-the-fly using the stored seeds, eliminating the need to store high-dimensional perturbation directions. This approach incurs only negligible overhead for storing K random seeds and scalar coefficients ($O(K)$, typically $K \ll d$); no full perturbation vector is materialized. The reconstruction therefore uses parameter-sized storage only for the gradient estimate itself.

Since these computations are performed within a `torch.no_grad()` context—without building a computational graph—the memory footprint is

$$\text{Mem}_0 = M_{\text{params}} + G_{\text{est}} + A_{\text{max}} + G_{\text{gradspool}},$$

where A_{max} denotes the activation memory of the largest layer. Consequently, the ZO peak memory does not accumulate activations across layers:

$$\text{Mem}_0 = 3d\beta + A_{\text{max}}.$$

For the evaluated configurations, A_{max} is smaller than the cumulative BP activation term.

A direct comparison shows that FO methods incur an additional activation memory term absent in ZO, implying $\text{Mem}_0 < \text{Mem}_1$ for all $S > 0$.

If gradient checkpointing is enabled for FO, the activation memory can be reduced from $O(L)$ to $O(\sqrt{L})$, yielding

$$\text{Mem}_1^{\text{ckpt}}(S) = 3d\beta + \sqrt{L}(HS^2 + SD_{\text{ffn}})\beta.$$

In this case, the memory advantage of ZO is somewhat diminished but still persists for sufficiently long sequences.

When FO uses a batch size $B > 1$, activation memory scales linearly with B , further increasing its memory footprint.

Under LoRA adaptation, both methods load the same base model parameters, but all gradient-related terms ($G_{\text{grads}}, G_{\text{gradspool}}$) scale with the

LoRA rank rather than d . Nevertheless, FO still requires storing activations and maintaining the computational graph for backpropagation, resulting in higher memory usage than ZO.

B.1.2 Step 2

In Step 2, both FOIF and ZOIF load the model parameters and the direction computed from Step 1. With an iterative optimizer, each FO gradient evaluation retains the BP graph, whereas its ZO counterpart uses transient forward activations. The peak-memory saving per gradient evaluation is therefore at least $A_{\text{activations}} - A_{\text{max}}$. Sequential evaluations add computation but do not, by themselves, multiply peak memory; implementations that retain higher-order graphs may incur additional FO memory. As B increases, both methods process more transient activations, while FOIF additionally retains the layer-wise activations required for BP.

If FO employs adaptive optimizers (e.g., Adam), additional optimizer states must be stored, further increasing memory consumption. These observations are consistent with findings in [Malladi et al. \(2023\)](#); [Zhang et al. \(2025\)](#).

Alternatively, if the IHVP is approximated via a diagonal Hessian instead of an optimizer, the approximation still relies on gradient estimation. In this case, FO again requires building a computational graph and caching activations, leading to higher memory usage than ZO.

B.1.3 Step 3

Step 3 exhibits the largest memory disparity between FOIF and ZOIF.

Both methods share the cost of loading model parameters and the parameter updates from Step 2. The key difference lies in how the influence score is computed:

- FOIF computes a separate gradient for each target sample, which entails storing activations, maintaining the computational graph, and caching gradient vectors.
- ZOIF requires only two forward passes per sample to estimate the influence, completely avoiding gradient computation and storage. It incurs only the activation memory of a single forward pass, i.e., A_{max} .

Combining the insights from Step 1, the per-sample memory saving is $d\beta + A_{\text{activations}} - A_{\text{max}}$.

In summary, ZOIF consistently incurs lower memory overhead than FOIF.

Step	FOIF Cost	ZOIF Cost	Ratio (ZO/FO)
Step 1	$N_{\text{prox}} \cdot 3C_{\text{fwd}}$	$N_{\text{prox}} \cdot (K + 1)C_{\text{fwd}}$	$\approx 10.3\times$
Step 2	$TB \cdot 3C_{\text{fwd}}$	$TB \cdot (K + 1)C_{\text{fwd}}$	$\approx 10.3\times$
Step 3	$N_{\text{pool}} \cdot 3C_{\text{fwd}}$	$N_{\text{pool}} \cdot 2C_{\text{fwd}}$	$\approx 0.67\times$

Table 3: Computational Complexity (FLOPs) Comparison ($K = 30$)

B.2 Computational Complexity Analysis

We provide a comprehensive analysis of the computational complexity (FLOPs) for both FOIF and ZOIF.

B.2.1 Theoretical Complexity Framework

We estimate FLOPs based on the cost of a single forward pass (C_{fwd}), where $C_{\text{fwd}} \approx 2 \cdot d \cdot S$ (with d as the number of parameters and S as the sequence length). Following standard conventions in LLM training, one backward pass costs approximately twice a forward pass ($C_{\text{bwd}} \approx 2C_{\text{fwd}}$), making the total cost of a first-order gradient update $C_{\text{grad}} \approx 3C_{\text{fwd}}$. Table 3 summarizes the complexity for each computational step, with $K = 30$ perturbations for ZOIF in our experiments.

B.2.2 Step-wise Analysis.

In Steps 1 and 2 (initialization stages), ZOIF requires K perturbations to estimate gradient and Hessian-vector product (HVP) directions. With $K = 30$, each iteration involves 31 forward passes, making it approximately **10.3** $\times$ more computationally expensive than FOIF’s single backpropagation ($3C_{\text{fwd}}$). This explains the increased runtime during early stages observed in our empirical evaluations.

However, in Step 3 (influence scoring over the large pool N_{pool}), ZOIF demonstrates an efficiency advantage: it requires only two forward passes per sample (for finite-difference loss estimation), whereas FOIF must perform a full backpropagation. Consequently, ZOIF reduces per-sample FLOPs by **33%** (from $3C_{\text{fwd}}$ to $2C_{\text{fwd}}$).

B.2.3 Scaling Behavior.

Both methods scale linearly with pool size, $O(N_{\text{pool}} \cdot d \cdot S)$. In Step 3, ZOIF has a lower per-sample constant and avoids activation accumulation across layers, which permits larger batches within a fixed memory budget. The complexity of ZOIF in Steps 1 and 2 scales linearly with K , while both methods scale linearly with model size d .

B.3 Time-Memory Trade-off

The theoretical FLOPs analysis reveals a deliberate **time-for-memory trade-off** inherent to ZOIF. While ZOIF incurs higher FLOPs in initialization steps, this computational overhead is the price paid for eliminating the $O(L)$ or $O(\sqrt{L})$ activation memory required for backpropagation. This trade-off enables influence-based selection on large models where the corresponding FOIF configuration would encounter Out-of-Memory (OOM) errors on identical hardware, making ZOIF a viable option under tight memory constraints.

B.4 Difference from Direct ZO Influence Computation

A direct ZO translation of conventional IF computation would reconstruct a random-direction gradient for every proxy and candidate sample. ZOIF instead exploits the shared structure of aggregate influence. Step 1 averages the proxy-side ZO gradients once, and Step 2 converts this average into a single shared direction $\hat{\mathbf{v}}$. The cost of these two steps is therefore amortized across the entire candidate pool.

The computational difference is explicit at the score level. A direct per-candidate ZO translation first reconstructs

$$\hat{\mathbf{g}}_i^{\text{dir}} = \frac{1}{K} \sum_{k=1}^K \frac{\ell(\boldsymbol{\theta}_0 + 2\alpha \mathbf{u}_k, z_i) - \ell(\boldsymbol{\theta}_0, z_i)}{2\alpha} \mathbf{u}_k, \\ \hat{\mathcal{I}}_i^{\text{dir}} = -m_p(\hat{\mathbf{g}}_i^{\text{dir}})^\top \hat{\mathbf{v}}, \quad (10)$$

which requires $(K+1)$ forward evaluations and a reconstructed gradient for each candidate. In contrast, ZOIF evaluates only the directional derivative

$$\hat{\mathcal{I}}_i^{\text{ZOIF}} = -m_p \frac{\ell(\boldsymbol{\theta}_0 + \alpha \hat{\mathbf{v}}, z_i) - \ell(\boldsymbol{\theta}_0, z_i)}{\alpha}. \quad (11)$$

Thus, candidate scoring uses exactly two forward evaluations, is independent of K , and never materializes a per-sample gradient vector.

Table 4 further distinguishes ZOIF from existing forward-oriented data-influence methods. In particular, it identifies where loss-only or ZO computation enters each method. ZOIF applies it throughout all three stages of the decomposed IF computation, whereas the alternatives use forward information only for a specific signal or scoring operation.

Unlike the alternatives in Table 4, ZOIF preserves the local Hessian-preconditioned IF target and uses ZO computation in all three stages while amortizing its proxy-side and curvature computation over the entire candidate pool. Its second stage

Table 4: Comparison of forward-oriented influence and data-valuation methods. The fourth column identifies the stage at which ZO or loss-only computation is used.

Method	Proxy/test-side construction	Candidate-side scoring signal	Placement of ZO/loss-only computation
Forward-INF (Ko et al., 2024)	Gradient ascent or continual updating on the test set	Loss change before and after the test-side update	Candidate scoring only; proxy/test-side updates still use gradients
Z0-Inf (Kokhlikyan et al., 2025)	Loss values from intermediate training checkpoints	Candidate loss evolution along the checkpoint trajectory	Loss-only scoring along an existing training trajectory
For-Value (Deng et al., 2026)	Final-layer representations and token-level prediction errors	Closed-form alignment of representations and prediction errors	Forward features and errors used directly for valuation
ZOIF (ours)	Averaged proxy ZO gradient followed by one shared direction	Directional loss difference in Eq. (11)	All three stages: ZO proxy gradient, ZO Step 2, and forward finite difference

is modular: the default ZOIF_opt solves the auxiliary problem with a ZO optimizer, while ZOIF_hvp, ZOIF_dia, and ZOIF_ide replace this solve with block-wise empirical-Fisher, diagonal, and identity approximations, respectively; all variants retain ZO estimation in the surrounding stages.

C Theoretical Results and Proofs

C.1 Notation for the Proofs

For the proofs, let

$$\begin{aligned} \mathbf{g}_{\text{proxy}} &= \frac{1}{m_p} \sum_{j=1}^{m_p} \nabla \ell(\boldsymbol{\theta}_0, z'_j), \\ \mathbf{H}_0 &= \nabla^2 \mathcal{L}(\boldsymbol{\theta}_0), \\ \Delta &= \boldsymbol{\theta}' - \boldsymbol{\theta}_0, \end{aligned}$$

and define

$$\begin{aligned} \mathcal{I}_{\text{FO}}(z_i) &= -m_p \nabla \ell(\boldsymbol{\theta}_0, z_i)^\top \mathbf{H}_0^{-1} \mathbf{g}_{\text{proxy}}, \\ \hat{\mathcal{I}}_{\text{ZOIF}}(z_i) &= w_i, \\ w_i &= -\frac{m_p}{\alpha} [\ell(\boldsymbol{\theta}_0 + \alpha \Delta, z_i) - \ell(\boldsymbol{\theta}_0, z_i)]. \end{aligned}$$

For an average over q samples with K independent Gaussian directions per sample, define

$$\begin{aligned} \mathcal{E}_d(q, K, \alpha, \delta) &= G \sqrt{\frac{d+1}{qK\delta}} \\ &+ \frac{\alpha M}{qK} \sum_{j=1}^q \sum_{k=1}^K \|\mathbf{u}_{jk}\|_2^3 \\ &+ \frac{2L_H \alpha^2}{3qK} \sum_{j=1}^q \sum_{k=1}^K \|\mathbf{u}_{jk}\|_2^4. \end{aligned} \quad (12)$$

Let $\mathbf{g}_S$ be the exact proxy-subset average, let $q_p = |S|$, and let $\mathbf{g}_{0,q_0}$ be the exact average over the q_0

reference samples used for $\tilde{\mathbf{g}}_0$. Define

$$\begin{aligned} \varepsilon_{\text{sub}} &= \|\mathbf{g}_S - \mathbf{g}_{\text{proxy}}\|_2, \\ \varepsilon_{\text{ref}} &= \|\mathbf{g}_{0,q_0} - \nabla \mathcal{L}(\boldsymbol{\theta}_0)\|_2, \\ E_p &= \varepsilon_{\text{sub}} + \mathcal{E}_d(q_p, K, \alpha, \delta_0/2), \\ E_0 &= \varepsilon_{\text{ref}} + \mathcal{E}_d(q_0, K, \alpha, \delta_0/2), \\ E_\tau &= E_p + E_0 + \tau. \end{aligned}$$

For the low-rank setting, write

$$\begin{aligned} \mathbf{H}_{0,r} &= \mathbf{U}_r^\top \mathbf{H}_0 \mathbf{U}_r, \\ \mathbf{H}_0^\dagger &= \mathbf{U}_r \mathbf{H}_{0,r}^{-1} \mathbf{U}_r^\top, \\ E_{r,\tau} &= \varepsilon_{\text{sub}} + \varepsilon_{\text{ref}} + \tau \\ &+ \mathcal{E}_r(q_p, K, \alpha, \delta_0/2) \\ &+ \mathcal{E}_r(q_0, K, \alpha, \delta_0/2), \end{aligned}$$

where $\mathcal{E}_r$ is Eq. (12) evaluated with d replaced by r , $\mathbf{u}_{jk} = \mathbf{U}_r \mathbf{q}_{jk}$, and $\mathbf{q}_{jk} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_r)$.

C.2 Proof of Theorem 1

Let $\mathbf{v}_{\text{FO}} = \mathbf{H}_0^{-1} \mathbf{g}_{\text{proxy}}$ and

$$d_i(\Delta) = \frac{\ell(\boldsymbol{\theta}_0 + \alpha \Delta, z_i) - \ell(\boldsymbol{\theta}_0, z_i)}{\alpha}.$$

By the Step 3 definition, $\hat{\mathcal{I}}_{\text{ZOIF}}(z_i) = -m_p d_i(\Delta)$. Define the directional finite-difference error by

$$\begin{aligned} \eta_i(\Delta) &= d_i(\Delta) \\ &- \nabla \ell(\boldsymbol{\theta}_0, z_i)^\top \Delta. \end{aligned}$$

Adding and subtracting the directional derivative along Δ gives

$$\begin{aligned} |\mathcal{I}_{\text{FO}}(z_i) - \hat{\mathcal{I}}_{\text{ZOIF}}(z_i)| &\leq m_p G \|\mathbf{v}_{\text{FO}} - \Delta\|_2 \\ &+ m_p |\eta_i(\Delta)|. \end{aligned}$$

We next decompose the direction error into ZO/solver and local-curvature contributions, and then bound the remaining directional finite-difference error.

C.2.1 Forward ZO Gradient Error

The estimator is the forward-difference Gaussian-smoothing construction of [Nesterov and Spokoiny \(2017\)](#), implemented with the random-direction updates used by [Malladi et al. \(2023\)](#). The following lemma records the finite-sample sketching and finite-radius terms needed here.

Lemma 3 (Forward ZO gradient error). *Under Assumptions 1–4, an average over q samples estimated with K independent Gaussian directions per sample satisfies, with probability at least $1 - \delta$,*

$$\|\tilde{\mathbf{g}} - \mathbf{g}_q\|_2 \leq \mathcal{E}_d(q, K, \alpha, \delta),$$

where $\mathbf{g}_q$ is the corresponding exact sample-average gradient.

Proof. For sample z_j , let $\mathbf{g}_j = \nabla \ell(\boldsymbol{\theta}_0, z_j)$. Taylor expansion of one forward quotient gives

$$\begin{aligned} & \frac{\ell(\boldsymbol{\theta}_0 + 2\alpha \mathbf{u}_{jk}, z_j) - \ell(\boldsymbol{\theta}_0, z_j)}{2\alpha} \\ &= \mathbf{g}_j^\top \mathbf{u}_{jk} + \alpha \mathbf{u}_{jk}^\top \nabla^2 \ell(\boldsymbol{\theta}_0, z_j) \mathbf{u}_{jk} + \rho_{jk}, \end{aligned}$$

where $|\rho_{jk}| \leq (2L_H \alpha^2/3) \|\mathbf{u}_{jk}\|_2^3$. After multiplication by $\mathbf{u}_{jk}$, the finite-radius error is at most

$$\alpha M \|\mathbf{u}_{jk}\|_2^3 + \frac{2L_H \alpha^2}{3} \|\mathbf{u}_{jk}\|_2^4.$$

For the random sketch, Gaussian moments give

$$\begin{aligned} \mathbb{E}[(\mathbf{u}\mathbf{u}^\top - \mathbf{I})\mathbf{g}] &= \mathbf{0}, \\ \mathbb{E}\|(\mathbf{u}\mathbf{u}^\top - \mathbf{I})\mathbf{g}\|_2^2 &= (d+1)\|\mathbf{g}\|_2^2. \end{aligned}$$

Independence across the qK directions and Assumption 2 therefore imply

$$\mathbb{E} \left\| \frac{1}{qK} \sum_{j=1}^q \sum_{k=1}^K (\mathbf{u}_{jk} \mathbf{u}_{jk}^\top - \mathbf{I}) \mathbf{g}_j \right\|_2^2 \leq \frac{(d+1)G^2}{qK}.$$

Markov's inequality applied to the squared norm bounds this component by $G\sqrt{(d+1)/(qK\delta)}$ with probability at least $1 - \delta$. Adding the realized forward-difference remainder proves Eq. (12). $\square$

C.2.2 Steps 1–2: ZO Estimation Error (Term A)

Apply Lemma 3 to the proxy subset and the reference subset. A union bound gives, with probability at least $1 - \delta_0$,

$$\begin{aligned} \|\tilde{\mathbf{g}}_{\text{proxy}} - \mathbf{g}_{\text{proxy}}\|_2 &\leq E_p, \\ \|\tilde{\mathbf{g}}_0 - \nabla \mathcal{L}(\boldsymbol{\theta}_0)\|_2 &\leq E_0. \end{aligned}$$

Define the signed estimation errors

$$\begin{aligned} e_p &= \tilde{\mathbf{g}}_{\text{proxy}} - \mathbf{g}_{\text{proxy}}, \\ e_0 &= \tilde{\mathbf{g}}_0 - \nabla \mathcal{L}(\boldsymbol{\theta}_0), \end{aligned}$$

and the Step 2 residual

$$\begin{aligned} \mathbf{r}_{\text{opt}} &= \nabla \mathcal{L}(\boldsymbol{\theta}_0 + \Delta) - \tilde{\mathbf{g}}_0 - \tilde{\mathbf{g}}_{\text{proxy}}, \\ \|\mathbf{r}_{\text{opt}}\|_2 &\leq \tau. \end{aligned}$$

The fundamental theorem of calculus gives the exact identity

$$\nabla \mathcal{L}(\boldsymbol{\theta}_0 + \Delta) - \nabla \mathcal{L}(\boldsymbol{\theta}_0) = \overline{\mathbf{H}}(\Delta)\Delta.$$

Consequently,

$$\overline{\mathbf{H}}(\Delta)\Delta = \mathbf{g}_{\text{proxy}} + e_p + e_0 + \mathbf{r}_{\text{opt}}.$$

Therefore, the direction error has the exact decomposition

$$\begin{aligned} \mathbf{v}_{\text{FO}} - \Delta &= (\mathbf{H}_0^{-1} - \overline{\mathbf{H}}(\Delta)^{-1})\mathbf{g}_{\text{proxy}} \\ &\quad - \overline{\mathbf{H}}(\Delta)^{-1}(e_p + e_0 + \mathbf{r}_{\text{opt}}). \end{aligned}$$

Combining this identity with the initial score bound yields the three error terms used in the proof:

$$\begin{aligned} & |\mathcal{I}_{\text{FO}}(z_i) - \hat{\mathcal{I}}_{\text{ZOIF}}(z_i)| \\ & \leq m_p G \underbrace{\|\overline{\mathbf{H}}(\Delta)^{-1}(e_p + e_0 + \mathbf{r}_{\text{opt}})\|_2}_{\text{Term A}} \\ & \quad + m_p G \underbrace{\|(\mathbf{H}_0^{-1} - \overline{\mathbf{H}}(\Delta)^{-1})\mathbf{g}_{\text{proxy}}\|_2}_{\text{Term B}} \\ & \quad + m_p \underbrace{|\eta_i(\Delta)|}_{\text{Term C}}. \end{aligned} \tag{13}$$

Term A captures the ZO gradient and finite-solver errors, Term B captures the local-curvature error, and Term C captures the Step 3 directional finite-difference error. We now bound each term separately. Assumption 3 yields

$$\|\Delta\|_2 \leq \frac{G + E_\tau}{\mu}, \tag{14}$$

$$\|\overline{\mathbf{H}}(\Delta)^{-1}(e_p + e_0 + \mathbf{r}_{\text{opt}})\|_2 \leq \frac{E_\tau}{\mu}. \tag{15}$$

Term A therefore includes the proxy and reference subsampling discrepancies, Gaussian sketching error, forward-difference bias, and the residual of the finite stochastic Step 2 solve. Exact local stationarity is the special case $\tau = 0$.

C.2.3 Step 2: Local Curvature Error (Term B)

Hessian Lipschitzness implies

$$\begin{aligned}\|\bar{\mathbf{H}}(\Delta) - \mathbf{H}_0\|_2 &\leq \int_0^1 L_H t \|\Delta\|_2 dt \\ &= \frac{L_H}{2} \|\Delta\|_2.\end{aligned}$$

The resolvent identity and Assumption 3 give

$$\begin{aligned}\|\mathbf{H}_0^{-1} - \bar{\mathbf{H}}(\Delta)^{-1}\|_2 &\leq \|\mathbf{H}_0^{-1}\|_2 \|\bar{\mathbf{H}}(\Delta) - \mathbf{H}_0\|_2 \|\bar{\mathbf{H}}(\Delta)^{-1}\|_2 \\ &\leq \frac{L_H}{2\mu^2} \|\Delta\|_2.\end{aligned}$$

Combining this result with Eq. (14) gives

$$\|(\mathbf{H}_0^{-1} - \bar{\mathbf{H}}(\Delta)^{-1})\mathbf{g}_{\text{proxy}}\|_2 \quad (16)$$

$$\leq \frac{L_H G(G + E_\tau)}{2\mu^3}. \quad (17)$$

This integrated-Hessian argument is exact and does not invoke a vector-valued mean-value theorem.

C.2.4 Step 3: Directional Finite-Difference Error (Term C)

Taylor's theorem with a Lipschitz-Hessian remainder gives

$$\begin{aligned}\ell(\boldsymbol{\theta}_0 + \alpha\Delta, z_i) &= \ell(\boldsymbol{\theta}_0, z_i) \\ &\quad + \alpha \nabla \ell(\boldsymbol{\theta}_0, z_i)^\top \Delta \\ &\quad + \frac{\alpha^2}{2} \Delta^\top \nabla^2 \ell(\boldsymbol{\theta}_0, z_i) \Delta \\ &\quad + R_i(\alpha),\end{aligned}$$

where $|R_i(\alpha)| \leq (L_H \alpha^3/6) \|\Delta\|_2^3$. Hence

$$\begin{aligned}|d_i(\Delta) - \nabla \ell(\boldsymbol{\theta}_0, z_i)^\top \Delta| &\leq \frac{M\alpha}{2} \|\Delta\|_2^2 + \frac{L_H \alpha^2}{6} \|\Delta\|_2^3 \\ &\leq \frac{M\alpha}{2} \left(\frac{G + E_\tau}{\mu} \right)^2 \\ &\quad + \frac{L_H \alpha^2}{6} \left(\frac{G + E_\tau}{\mu} \right)^3.\end{aligned}$$

Substitution of the three bounds into Eq. (13) proves Theorem 1.

C.3 Finite Step 2 Runs

Theorem 1 is stated directly for a finite Step 2 output through τ . A local stationary point sets $\tau = 0$; a stopped stochastic run remains covered when its

full-objective residual is bounded by $\tau > 0$. The result is therefore conditional on the quality of the returned local point and does not claim that every task, dataset, or parameter choice makes the ZO solver converge. If the norm projection is active, the analogous statement uses a projected gradient-mapping residual and must additionally account for the realized projection effect. Under analogous local effective-rank conditions, Malladi et al. (2023, Theorem 1 and Lemma 3) show that the per-step and global convergence rates of ZO-SGD can depend on local effective rank rather than ambient dimension. Their setting differs from our auxiliary problem, so this result supports the plausibility of Step 2 without supplying its residual bound.

C.4 Proof of Theorem 2

Write $\Delta = \mathbf{U}_r \boldsymbol{\xi}$, $\mathbf{g}_{\text{proxy},r} = \mathbf{U}_r^\top \mathbf{g}_{\text{proxy}}$, and $\bar{\mathbf{H}}_r(\boldsymbol{\xi}) = \mathbf{U}_r^\top \bar{\mathbf{H}}(\Delta) \mathbf{U}_r$. Projecting the fundamental-theorem-of-calculus identity used above gives

$$\bar{\mathbf{H}}_r(\boldsymbol{\xi}) \boldsymbol{\xi} = \mathbf{g}_{\text{proxy},r} + \mathbf{e}_{p,r} + \mathbf{e}_{0,r} + \mathbf{r}_{\text{opt},r}.$$

Here $\mathbf{e}_{p,r} = \mathbf{U}_r^\top \mathbf{e}_p$, $\mathbf{e}_{0,r} = \mathbf{U}_r^\top \mathbf{e}_0$, and $\mathbf{r}_{\text{opt},r} = \mathbf{U}_r^\top \mathbf{r}_{\text{opt}}$. The projected curvature condition bounds both $\|\mathbf{H}_{0,r}^{-1}\|_2$ and $\|\bar{\mathbf{H}}_r(\boldsymbol{\xi})^{-1}\|_2$ by $1/\mu$. Thus, the decomposition for Terms A and B applies verbatim in the projected coordinates. The full-space subsampling bounds remain valid because multiplication by $\mathbf{U}_r^\top$ is nonexpansive.

For the ZO terms, let $\mathbf{q}_{jk} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_r)$ and $\mathbf{u}_{jk} = \mathbf{U}_r \mathbf{q}_{jk}$. Orthogonality gives $\|\mathbf{u}_{jk}\|_2 = \|\mathbf{q}_{jk}\|_2$, and the projected Gaussian-sketch identity becomes

$$\mathbb{E}\|(\mathbf{q}\mathbf{q}^\top - \mathbf{I}_r)\mathbf{g}_r\|_2^2 = (r+1)\|\mathbf{g}_r\|_2^2.$$

Hence Eq. (12) holds with d replaced by r . Finally, $\|\Delta\|_2 = \|\boldsymbol{\xi}\|_2$, so the Step 3 Taylor bound is unchanged. These three observations yield Theorem 2 with E_τ replaced by $E_{r,\tau}$. The target direction in the original coordinates is

$$\mathbf{H}_0^\dagger \mathbf{g}_{\text{proxy}} = \mathbf{U}_r \mathbf{H}_{0,r}^{-1} \mathbf{U}_r^\top \mathbf{g}_{\text{proxy}}.$$

Comparison with the unrestricted full-space FOIF requires adding the norm of the omitted projection component; low effective rank alone does not make this bias zero.

C.5 Unified Definitions of the Four Step 2 Variants

Let $\tilde{\mathbf{g}}_{q,b}$ denote the ZO gradient estimate for reference-pool sample q in parameter block b , and

let N be the size of the Step 2 subset. All sample gradients below use the forward estimator analyzed above. The four reported variants are:

- ZOIF_opt: for a minibatch loss $\mathcal{L}_B$, the stochastic auxiliary objective is

$$J_B(\delta) = \mathcal{L}_B(\theta_0 + \delta) - \mathcal{L}_B(\theta_0) - \langle \tilde{\mathbf{g}}_{0,B}, \delta \rangle - \langle \tilde{\mathbf{g}}_{\text{proxy}}, \delta \rangle.$$

Both $\tilde{\mathbf{g}}_{0,B}$ and the update direction for J_B are estimated by ZOGE, followed by projection onto $\|\delta\|_2 \leq 0.05$. Theorem 1 gives the local bound when the projection is inactive; an active projection requires the projected residual described above.

- ZOIF_hvp: this variant approximates the curvature-aware direction using a DataInf-style damped block-wise empirical-Fisher inverse. For block b , set

$$\lambda_b = \frac{1}{10N} \sum_{q=1}^N \text{mean}(\tilde{\mathbf{g}}_{q,b}^{\odot 2}).$$

The closed-form operator is

$$\hat{\mathbf{B}}_{\text{hvp},b} = \frac{1}{N\lambda_b} \sum_{q=1}^N \left(\mathbf{I} - \frac{\tilde{\mathbf{g}}_{q,b} \tilde{\mathbf{g}}_{q,b}^\top}{\lambda_b + \|\tilde{\mathbf{g}}_{q,b}\|_2^2} \right).$$

For $\lambda_b > 0$, $\|\hat{\mathbf{B}}_{\text{hvp},b}\|_2 \leq 1/\lambda_b$.

- ZOIF_dia: the implementation forms the coordinate-wise empirical second moment

$$\hat{\mathbf{h}} = \frac{1}{N} \sum_{q=1}^N \tilde{\mathbf{g}}_q^{\odot 2}$$

and uses $\hat{\mathbf{B}}_{\text{dia}} = \text{Diag}(\hat{\mathbf{h}} + 10^{-8}\mathbf{1})^{-1}$. Hence $\|\hat{\mathbf{B}}_{\text{dia}}\|_2 \leq 10^8$. This construction adopts the diagonal-approximation principle used by methods such as SOUL (Jia et al., 2024). In the forward-only setting, the empirical second moment of the ZO gradient supplies the diagonal proxy.

- ZOIF_ide: $\hat{\mathbf{B}}_{\text{ide}} = \mathbf{I}$, so $\hat{\mathbf{v}}_{\text{ide}} = \tilde{\mathbf{g}}_{\text{proxy}}$.

C.6 Theoretical Extensions to Plug-in Step 2 Operators

The main theorem analyzes Algorithm 1. For a plug-in Step 2 variant $a \in \{\text{hvp}, \text{dia}, \text{ide}\}$, suppose $\mathbf{H}_{0,\lambda} = \mathbf{H}_0 + \lambda \mathbf{I}$ is invertible, and let

$\mathbf{v}_\lambda = \mathbf{H}_{0,\lambda}^{-1} \mathbf{g}_{\text{proxy}}$, $\mathcal{I}_\lambda(z_i) = -m_p \nabla \ell(\theta_0, z_i)^\top \mathbf{v}_\lambda$, and $\hat{\mathbf{v}}_a = \hat{\mathbf{B}}_a \tilde{\mathbf{g}}_{\text{proxy}}$. Define

$$\rho_a = \|(\mathbf{H}_{0,\lambda}^{-1} - \hat{\mathbf{B}}_a) \mathbf{g}_{\text{proxy}}\|_2.$$

Then

$$\|\mathbf{v}_\lambda - \hat{\mathbf{v}}_a\|_2 \leq \rho_a + \|\hat{\mathbf{B}}_a\|_2 \cdot \|\tilde{\mathbf{g}}_{\text{proxy}} - \mathbf{g}_{\text{proxy}}\|_2. \quad (18)$$

Proposition 4. *If the Step 3 segment satisfies Assumptions 1, 2, and 4, define*

$$\hat{\mathcal{I}}_a(z_i) = -m_p \frac{\ell(\theta_0 + \alpha \hat{\mathbf{v}}_a, z_i) - \ell(\theta_0, z_i)}{\alpha}.$$

Then

$$\begin{aligned} |\mathcal{I}_\lambda(z_i) - \hat{\mathcal{I}}_a(z_i)| &\leq m_p G \|\mathbf{v}_\lambda - \hat{\mathbf{v}}_a\|_2 \\ &\quad + \frac{m_p M \alpha}{2} \|\hat{\mathbf{v}}_a\|_2^2 \\ &\quad + \frac{m_p L_H \alpha^2}{6} \|\hat{\mathbf{v}}_a\|_2^3. \end{aligned}$$

Proof. Adding and subtracting $\hat{\mathbf{B}}_a \mathbf{g}_{\text{proxy}}$ gives

$$\begin{aligned} \mathbf{v}_\lambda - \hat{\mathbf{v}}_a &= (\mathbf{H}_{0,\lambda}^{-1} - \hat{\mathbf{B}}_a) \mathbf{g}_{\text{proxy}} \\ &\quad + \hat{\mathbf{B}}_a (\mathbf{g}_{\text{proxy}} - \tilde{\mathbf{g}}_{\text{proxy}}), \end{aligned}$$

which proves Eq. (18). Next, add and subtract $-m_p \nabla \ell(\theta_0, z_i)^\top \hat{\mathbf{v}}_a$ between $\mathcal{I}_\lambda(z_i)$ and $\hat{\mathcal{I}}_a(z_i)$. Cauchy–Schwarz bounds the resulting direction-error term by $m_p G \|\mathbf{v}_\lambda - \hat{\mathbf{v}}_a\|_2$. The same Lipschitz-Hessian Taylor expansion used for Term C bounds the remaining forward-difference term by the final two terms in the proposition. $\square$

Equation (18) separates ZO gradient-estimation error from each variant’s structural operator mismatch; this extension is separate from the Algorithm 1 theorem.

D Supplementary Experimental Details

D.1 Datasets and Evaluation

Our evaluation code framework follows that of Iverson et al. (2025).

MMLU (Hendrycks et al., 2021): We use the official validation split as proxy data and the official test split for evaluation. Model performance is measured using Average Accuracy. During evaluation, we use a 5-shot setting, consistent with Iverson et al. (2025).

BBH (Big-Bench Hard) (Suzgun et al., 2022): This benchmark comprises 27 challenging reasoning tasks across diverse domains. For each task, we randomly sample 1/10 of the data as proxy data for influence computation, with the remaining 9/10 used as test data. Evaluation is based on Average Accuracy. Evaluation follows a 3-shot setting, as in Iverson et al. (2025).

SQuAD (Rajpurkar et al., 2016): A reading comprehension dataset with official training and validation splits. We sample 1,174 examples from the training split as proxy data and use the official validation split for evaluation. Following Iverson et al. (2025), model performance is assessed using the F1 score, and no in-context examples are used during evaluation on SQuAD.

All evaluation metrics were computed using the corresponding functions from the Hugging Face evaluate library.

All datasets are converted into a common evaluation format. Their proxy-to-test sizes are approximately 1:9, and the same fixed splits are used for all compared methods to ensure a fair comparison. The number of proxy samples used for IFs computation and the number of test samples used for final performance evaluation are summarized in Table 5.

Dataset	$ \mathcal{D}_{\text{proxy}} $	$ \mathcal{D}_{\text{test}} $
MMLU	1,531	14,042
BBH	649	5,862
SQuAD	1,174	10,570

Table 5: Sample sizes for the proxy and test splits of each dataset.

D.2 Implementation and Hyperparameter Settings

Influence functions are computed in bfloat16 for all models. On Llama and Qwen, we use LoRA (rank 16, scaling 64) for IFs computation and switch to a higher-capacity LoRA (rank 64, scaling 128) during fine-tuning. For pythia-70m, we perform full-parameter gradient computation and fine-tuning.

In Step 1, since the proxy dataset sizes for all three datasets are relatively small, we set $|\mathcal{S}| = m_p$ to ensure estimation accuracy.

When computing IFs, ZOIF_hvp follows the block-wise empirical-Fisher inverse approximation of Kwon et al. (2024). To reduce computation, we draw a random subset of 2,000 examples from the data pool for Step 2. The implementation sets

each block damping value to one tenth of the mean coordinate-wise ZO gradient second moment over this subset. ZOIF_dia uses the same ZO gradients to form a coordinate-wise empirical second moment with a 10^{-8} stabilizer. For a fair comparison, ZOIF_dia, ZOIF_opt, and their matched FOIF counterparts use the same subset.

When computing Step 2 with our ZOIF_opt method, we employ a ZO optimizer (Malladi et al., 2023) to save memory. Since we approximate the IHVP using $\Delta\theta$, and parameter updates remain within a small range, we use a small batch size of 8 and 50 optimization steps. To prevent gradient explosion, we apply gradient clipping with a threshold of 0.05. For pythia-70m, we set the learning rate to 1e-6. For all other models, which optimize only the LoRA layers, the learning rate is set to 1e-4. In contrast, FOIF_opt uses exact first-order gradients for gradient descent, with all other hyperparameters kept identical to those of ZOIF_opt.

For memory and runtime comparison, we use the MMLU proxy set in Step 1, a subset of 2,000 examples drawn from the data pool in Step 2, and the full data pool in Step 3. The pythia-70m model is relatively small and sensitive to fluctuations caused by data tokenization. To mitigate this variability in the memory comparison for Step 1 and 2, we construct a synthetic dataset by duplicating a single training example.

For fine-tuning, we use the AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate of 2e-4 and fine-tune the Llama and Qwen series models for three epochs. Given that the pythia-70m model is significantly smaller, we use a smaller learning rate of 1e-6 and train it for one epoch. Due to GPU memory constraints, the per-device batch size is set to 1, combined with gradient accumulation over 8 steps to achieve an effective batch size of 8. FlashAttention-2 is enabled when fine-tuning the Llama and Qwen series models and disabled for pythia-70m; model checkpoint saving is disabled to reduce storage overhead.

D.3 Details of Baseline Methods

- FOIF: The standard first-order influence function method. For fair comparison, we implement first-order counterparts corresponding exactly to our ZO variants, namely FOIF_opt, FOIF_hvp, FOIF_dia, and FOIF_ide.
- LESS (Xia et al., 2024) and TRAK (Park et al.,

2023): Gradient-based influence estimation methods that use gradient projection and random sketching for scalable data attribution. Both require BP to compute the gradient features used by their scores.

- EMB: A data selection method based on embedding-space similarity (Ni et al., 2022; Lee et al., 2024). We use the embedding-gemma-300M model from Vera et al. (2025) to compute embeddings for each sample, then calculate the cosine similarity between each candidate sample and the average embedding of the proxy set. Samples are ranked from highest to lowest similarity, and the top 10000 are selected.
- PPL: A perplexity-based data selection strategy (Yin and Rush, 2025). We compute the perplexity of each candidate sample using the target LLM and sort the samples in descending order of perplexity. Following Yin and Rush (2025), we consider two sampling schemes: Top_PPL, which selects the top 10,000 samples with the highest perplexity, and Mid_PPL, which selects 10,000 samples from the middle of the perplexity distribution.
- Random: Randomly samples 10000 examples from the candidate pool for fine-tuning.
- Orig: The performance of the original, un-fine-tuned model, serving as a lower-bound reference.

D.4 Hardware and Reproducibility

All reported experimental results are averaged over three independent runs.

Primary model fine-tuning and downstream evaluations were conducted on NVIDIA A100 GPUs with 80GB memory, using two Hygon C86 7390 32-core processors, Ubuntu 22.04, Python 3.12, PyTorch 2.7.0, and CUDA 11.8. For rigorous efficiency cross-validation, we additionally used a domestic heterogeneous AI accelerator with 64GB device memory. This environment used one Hygon C86 7185 32-core processor (6 CPU cores allocated), 115GB system memory, Ubuntu 22.04, Python 3.11, PyTorch 2.9.0, and DTK 26.04. While absolute memory and runtime metrics varied slightly across hardware platforms (± 0.02 GB memory, $\sim 1.5\times$ speedup on A100), the relative efficiency advantages of ZOIF over FOIF remained

consistent, confirming hardware-agnostic scalability.

E Supplementary Experimental Results

E.1 Comparison of Memory and Time Efficiency

E.1.1 Memory Efficiency

The peak GPU memory results for all three steps are shown in Table 6, demonstrating similar findings. In particular, ZOIF_opt exhibits a clear advantage over FOIF_opt in terms of GPU memory usage. Gradient-based methods such as LESS and TRAK require backpropagation and storage of per-sample gradient features. On larger models, this memory overhead exceeded our hardware budget; consequently, we could not run these methods on Llama-2-7B or Qwen3-1.7B. This limitation further motivates forward-only influence estimation for memory-constrained settings.

E.1.2 Time Efficiency

Table 7 presents a step-by-step breakdown of the runtime. Although the zeroth-order (ZO) perturbation leads to increased computational overhead in the earlier steps for ZOIF, its overall runtime remains on par with that of FOIF.

E.1.3 Comparison with Memory Optimization Techniques

First-order methods such as FOIF can reduce activation memory using Gradient Checkpointing (GC) and Activation Offloading. We evaluate these techniques on Llama-3.2-1B ($K = 30$). As shown in Table 8, applying both techniques reduces the peak memory of FOIF from 3.91 GB to 3.18 GB, which remains **7.4% higher** than the 2.96 GB required by ZOIF. The optimizations also increase the runtime of FOIF relative to its unoptimized setting: GC, offloading, and their combination require 0.125h, 0.750h, and 0.167h, respectively, compared with 0.058h for the unoptimized method.

These Step 1 results show that ZOIF retains a lower measured memory footprint than the evaluated first-order configurations, including those using gradient checkpointing and activation offloading.

E.2 Comparison of Fine-tuning Performance Across Different Data Selection Methods

E.2.1 Results on pythia-70m

Results for pythia-70m fine-tuned using data selected under Strategy 1 are shown in Table 9.

LLM	FOIF					ZOIF					Notes
	Step 1	Step 2			Step 3	Step 1	Step 2			Step 3	
	FOIF_dia	FOIF_hvp	FOIF_opt			ZOIF_dia	ZOIF_hvp	ZOIF_opt			
pythia-70m	0.68	0.94	0.86	2.70	0.92	0.63	0.90	0.63	1.72	0.51	FULL
Llama-3.2-1B	3.91	3.94	3.96	10.67	3.91	2.96	2.97	2.97	3.00	2.95	LORA
Llama-2-7B	15.74	15.56	15.62	34.39	15.46	13.08	13.15	13.15	13.27	13.02	LORA
Qwen3-0.6B	3.09	3.11	3.13	15.77	3.10	1.92	1.94	1.94	1.98	1.91	LORA
Qwen3-1.7B	5.68	5.73	5.74	21.46	5.69	4.03	4.06	4.06	4.11	4.01	LORA

Table 6: Peak GPU memory usage (in GB) for FOIF and ZOIF across all steps.

LLM	FOIF					ZOIF					Notes
	Step 1	Step 2			Step 3	Step 1	Step 2			Step 3	
	FOIF_dia	FOIF_hvp	FOIF_opt			ZOIF_dia	ZOIF_hvp	ZOIF_opt			
pythia-70m	0.017	0.021	0.040	0.033	37.60	0.27	0.33	0.58	0.60	22.50	FULL
Llama-3.2-1B	0.058	0.083	0.175	0.050	34.00	0.80	0.92	2.28	0.80	30.00	LORA
Llama-2-7B	0.100	0.150	0.320	0.050	43.00	2.00	2.50	5.00	2.00	40.00	LORA
Qwen3-0.6B	0.100	0.150	0.310	0.050	40.00	1.50	1.70	3.50	1.00	31.75	LORA
Qwen3-1.7B	0.100	0.150	0.320	0.050	40.30	1.50	1.80	3.50	1.60	32.00	LORA

Table 7: Computation time (in hours) for FOIF and ZOIF across all steps.

Method	Memory (GB)	Time (h)
FOIF	3.91	0.058
ZOIF	2.96	0.800
FOIF+GC	3.21	0.125
FOIF+Offload	3.19	0.750
FOIF+Both	3.18	0.167

Table 8: Step 1 memory and runtime on Llama-3.2-1B ($K = 30$).

Limited by the inherent capacity of pythia-70m, fine-tuning yields only marginal improvements in model performance. Nevertheless, we observe that ZOIF_hvp achieves the highest average score and slightly outperforms the other methods. Moreover, the ZOIF variants as a whole demonstrate performance comparable to that of the FOIF approaches. The Strategy 2 results are shown in Table 10 and exhibit a similar pattern.

E.2.2 Results on Llama

Llama-3.2-1B (Strategy 1). Evaluation results for Llama-3.2-1B under Strategy 1 are shown in Table 11. TRAK achieves the highest average score (45.73), followed by LESS (44.53), with ZOIF_hvp attaining 43.98—within 4% of the best result. Crucially, both TRAK and LESS require backpropagation through the full computational graph, incurring activation memory that scales with model depth and sequence length. By contrast, ZOIF operates entirely via forward passes, achieving competitive downstream performance at a fraction of the mem-

Method	MMLU Acc.	BBH Acc.	SQuAD F1	Macro Avg.
ZOIF_dia	23.45	7.24	0.64	10.44 $\pm$ 0.01
ZOIF_hvp	23.17	9.25	0.65	11.03 $\pm$ 0.00
ZOIF_ide	23.38	7.38	0.66	10.47 $\pm$ 0.01
ZOIF_opt	23.30	6.31	0.67	10.09 $\pm$ 0.01
FOIF_dia	23.29	7.03	0.59	10.30 $\pm$ 0.01
FOIF_hvp	23.60	7.09	0.54	10.41 $\pm$ 0.02
FOIF_ide	23.26	7.39	0.67	10.44 $\pm$ 0.00
FOIF_opt	23.36	7.63	0.68	10.55 $\pm$ 0.01
EMB	23.09	6.08	0.52	9.90 $\pm$ 0.01
Top_PPL	23.16	6.57	0.65	10.13 $\pm$ 0.01
Mid_PPL	23.18	6.86	0.65	10.23 $\pm$ 0.01
Random	23.17	7.58	0.61	10.46 $\pm$ 0.00
Orig	22.95	1.44	0.58	8.32

Table 9: Held-out test performance on pythia-70m (Strategy 1). The macro-average entry is reported as mean $\pm$ standard deviation over three independent runs.

Method	MMLU Acc.	BBH Acc.	SQuAD F1	Macro Avg.
ZOIF_dia	23.29	7.09	0.70	10.36 $\pm$ 0.02
ZOIF_hvp	23.29	7.45	0.64	10.46 $\pm$ 0.01
ZOIF_ide	23.35	8.59	0.55	10.83 $\pm$ 0.01
ZOIF_opt	23.55	6.51	0.57	10.21 $\pm$ 0.02
FOIF_dia	23.48	6.28	0.61	10.12 $\pm$ 0.01
FOIF_hvp	23.19	7.14	0.62	10.32 $\pm$ 0.03
FOIF_ide	23.33	7.57	0.63	10.51 $\pm$ 0.01
FOIF_opt	23.63	9.51	0.56	11.23 $\pm$ 0.00
EMB	23.36	6.73	0.63	10.24 $\pm$ 0.01
Top_PPL	23.17	6.57	0.65	10.13 $\pm$ 0.01
Mid_PPL	23.18	6.87	0.65	10.23 $\pm$ 0.01
Random	23.17	7.58	0.61	10.46 $\pm$ 0.00
Orig	22.95	1.44	0.58	8.32

Table 10: Held-out test performance on pythia-70m (Strategy 2). The macro-average entry is reported as mean $\pm$ standard deviation over three independent runs.

ory cost. This illustrates the central advantage of ZOIF: it offers a practical, memory-efficient alternative for influence-based data selection, delivering performance close to gradient-based methods without the prohibitive memory demands. Furthermore, ZOIF consistently outperforms non-gradient baselines (EMB, PPL, Random) by a clear margin.

Method	MMLU Acc.	BBH Acc.	SQuAD F1	Macro Avg.
ZOIF_dia	34.36	30.81	63.21	42.79 $\pm$ 0.92
ZOIF_hvp	34.23	29.63	68.09	43.98 $\pm$ 0.65
ZOIF_ide	33.85	30.57	64.27	42.90 $\pm$ 0.71
ZOIF_opt	31.37	30.77	65.80	42.64 $\pm$ 1.59
FOIF_dia	30.59	29.92	63.31	41.27 $\pm$ 1.23
FOIF_hvp	31.11	30.93	59.50	40.51 $\pm$ 1.38
FOIF_ide	32.43	30.55	60.85	41.28 $\pm$ 1.59
FOIF_opt	30.77	30.20	60.03	40.34 $\pm$ 1.32
LESS	32.98	28.99	71.62	44.53 $\pm$ 0.92
TRAK	35.65	29.38	72.17	45.73 $\pm$ 1.07
EMB	34.49	30.70	59.21	41.47 $\pm$ 0.31
Top_PPL	31.98	28.05	57.39	39.14 $\pm$ 0.73
Mid_PPL	32.47	28.27	57.78	39.51 $\pm$ 0.60
Random	32.79	30.58	62.31	41.89 $\pm$ 0.52
Orig	30.69	28.92	47.42	35.67

Table 11: Held-out test performance on Llama-3.2-1B (Strategy 1). The average is the unweighted macro-average of MMLU accuracy, BBH accuracy, and SQuAD F1. The macro-average entry is reported as mean $\pm$ standard deviation over three independent runs.

Results for Llama-3.2-1B fine-tuned using data selected under Strategy 2 are shown in Table 12. Under this strategy, TRAK again achieves the highest average score, while ZOIF variants deliver compet-

itive performance consistent with the main-text observations, reinforcing the memory-performance trade-off. For Llama-2-7B, results under Strategy 1 and Strategy 2 are reported in Tables 13 and 14, respectively, showing that ZOIF remains on par with FOIF.

Method	MMLU Acc.	BBH Acc.	SQuAD F1	Macro Avg.
ZOIF_dia	33.05	30.31	66.40	43.25 $\pm$ 1.04
ZOIF_hvp	33.17	31.15	58.45	40.92 $\pm$ 2.08
ZOIF_ide	33.05	30.34	63.17	42.19 $\pm$ 1.13
ZOIF_opt	31.33	31.26	65.08	42.55 $\pm$ 0.89
FOIF_dia	33.69	30.86	65.61	43.39 $\pm$ 1.73
FOIF_hvp	32.14	30.44	64.11	42.23 $\pm$ 1.19
FOIF_ide	31.51	30.86	68.49	43.62 $\pm$ 1.48
FOIF_opt	33.99	30.85	56.24	40.36 $\pm$ 0.66
LESS	33.04	30.87	64.35	42.75 $\pm$ 1.21
TRAK	36.26	30.69	73.92	46.96 $\pm$ 1.00
EMB	34.70	30.58	69.44	44.91 $\pm$ 0.63
Top_PPL	32.96	28.73	59.18	40.29 $\pm$ 1.01
Mid_PPL	31.82	28.58	57.48	39.29 $\pm$ 0.66
Random	32.23	30.71	62.11	41.68 $\pm$ 0.52
Orig	30.69	28.92	47.42	35.67

Table 12: Held-out test performance on Llama-3.2-1B (Strategy 2). The macro-average entry is reported as mean $\pm$ standard deviation over three independent runs.

E.2.3 Results on Qwen

Fine-tuning and evaluation results for Qwen3-0.6B (Strategy 2) and Qwen3-1.7B (Strategies 1 and 2) are reported in Tables 15, 16, and 17. For Qwen3-0.6B (Strategy 2), ZOIF_opt achieves the highest average score (51.72), outperforming LESS (50.16) and yielding a marginally higher average than TRAK (51.67). On Qwen3-1.7B, ZOIF consistently matches FOIF in overall performance, demonstrating robust results across model scales and strategies.

E.3 Hyperparameter Sensitivity Analysis

E.3.1 Sensitivity of K and α

The results are presented in Table 18. In this sensitivity setting, the best observed performance occurs at $\alpha = 0.01$ for all four variants. A larger α can amplify truncation error due to the first-order approximation, whereas an excessively small α can make gradient estimation more vulnerable to floating-point noise and numerical instability. This result is specific to the evaluated Llama-3.2-1B

Method	MMLU Acc.	BBH Acc.	SQuAD F1	Macro Avg.
ZOIF_dia	25.29	35.15	79.70	46.71 $\pm$ 0.50
ZOIF_hvp	25.45	36.16	82.86	48.16 $\pm$ 0.77
ZOIF_ide	26.22	35.57	83.58	48.46 $\pm$ 1.00
ZOIF_opt	25.93	35.24	83.33	48.17 $\pm$ 0.61
FOIF_dia	25.53	36.31	82.84	48.23 $\pm$ 0.35
FOIF_hvp	25.36	36.84	82.50	48.24 $\pm$ 0.98
FOIF_ide	25.85	35.27	81.84	47.65 $\pm$ 0.96
FOIF_opt	25.11	35.37	77.58	46.02 $\pm$ 0.79
EMB	25.66	37.14	80.91	47.90 $\pm$ 0.49
Top_PPL	25.30	32.68	76.15	44.71 $\pm$ 0.32
Mid_PPL	25.42	33.70	72.95	44.02 $\pm$ 0.18
Random	25.40	36.63	78.86	46.96 $\pm$ 0.36
Orig	25.24	32.26	42.87	33.46

Table 13: Held-out test performance on Llama-2-7B (Strategy 1). The macro-average entry is reported as mean $\pm$ standard deviation over three independent runs.

Method	MMLU Acc.	BBH Acc.	SQuAD F1	Macro Avg.
ZOIF_dia	25.77	35.85	80.32	47.31 $\pm$ 0.32
ZOIF_hvp	25.51	34.89	83.18	47.86 $\pm$ 0.36
ZOIF_ide	25.06	35.02	69.44	43.17 $\pm$ 0.81
ZOIF_opt	25.60	35.16	82.49	47.75 $\pm$ 1.32
FOIF_dia	25.73	33.81	83.81	47.78 $\pm$ 0.46
FOIF_hvp	25.39	34.37	82.49	47.42 $\pm$ 0.11
FOIF_ide	25.16	34.98	84.54	48.22 $\pm$ 0.96
FOIF_opt	25.42	34.39	75.78	45.20 $\pm$ 0.89
EMB	25.26	36.06	84.09	48.47 $\pm$ 0.20
Top_PPL	25.23	32.91	75.88	44.67 $\pm$ 0.32
Mid_PPL	25.39	33.74	72.71	43.95 $\pm$ 0.18
Random	25.41	36.39	79.00	46.93 $\pm$ 0.36
Orig	25.24	32.26	42.87	33.46

Table 14: Held-out test performance on Llama-2-7B (Strategy 2). The macro-average entry is reported as mean $\pm$ standard deviation over three independent runs.

Method	MMLU Acc.	BBH Acc.	SQuAD F1	Macro Avg.
ZOIF_dia	43.32	35.06	75.55	51.31 $\pm$ 0.45
ZOIF_hvp	44.33	33.13	76.05	51.17 $\pm$ 0.81
ZOIF_ide	44.21	34.64	72.78	50.54 $\pm$ 0.94
ZOIF_opt	45.12	34.53	75.52	51.72 $\pm$ 0.65
FOIF_dia	43.04	34.26	73.91	50.41 $\pm$ 0.60
FOIF_hvp	44.35	33.96	70.88	49.73 $\pm$ 1.15
FOIF_ide	44.47	34.85	73.23	50.85 $\pm$ 0.40
FOIF_opt	42.89	33.70	75.53	50.71 $\pm$ 0.93
LESS	43.15	33.22	74.11	50.16 $\pm$ 0.31
TRAK	44.98	33.42	76.60	51.67 $\pm$ 0.49
EMB	45.65	35.21	73.59	51.48 $\pm$ 0.16
Top_PPL	45.75	34.00	71.52	50.43 $\pm$ 0.27
Mid_PPL	45.82	33.83	72.31	50.66 $\pm$ 0.24
Random	45.27	34.33	69.51	49.70 $\pm$ 0.37
Orig	47.09	30.54	48.45	42.03

Table 15: Held-out test performance on Qwen3-0.6B (Strategy 2). The macro-average entry is reported as mean $\pm$ standard deviation over three independent runs.

Method	MMLU Acc.	BBH Acc.	SQuAD F1	Macro Avg.
ZOIF_dia	57.55	41.46	82.49	60.50 $\pm$ 0.37
ZOIF_hvp	57.38	41.85	80.16	59.80 $\pm$ 0.28
ZOIF_ide	57.68	42.75	81.08	60.50 $\pm$ 0.61
ZOIF_opt	57.36	41.78	83.20	60.78 $\pm$ 0.16
FOIF_dia	58.25	42.37	83.99	61.53 $\pm$ 0.59
FOIF_hvp	58.01	42.25	82.58	60.95 $\pm$ 1.06
FOIF_ide	58.54	41.66	81.43	60.54 $\pm$ 1.04
FOIF_opt	57.81	40.33	82.30	60.15 $\pm$ 0.65
EMB	58.34	42.37	83.04	61.25 $\pm$ 0.33
Top_PPL	57.59	41.94	82.70	60.74 $\pm$ 0.65
Mid_PPL	58.59	41.11	81.57	60.42 $\pm$ 0.56
Random	58.85	42.42	81.93	61.07 $\pm$ 0.44
Orig	59.88	40.50	40.66	47.01

Table 16: Held-out test performance on Qwen3-1.7B (Strategy 1). The macro-average entry is reported as mean $\pm$ standard deviation over three independent runs.

Method	MMLU Acc.	BBH Acc.	SQuAD F1	Macro Avg.
ZOIF_dia	57.81	41.51	82.25	60.52 $\pm$ 0.40
ZOIF_hvp	57.60	41.08	83.60	60.76 $\pm$ 0.45
ZOIF_ide	57.15	41.04	81.81	60.00 $\pm$ 1.17
ZOIF_opt	57.84	41.34	82.52	60.57 $\pm$ 0.47
FOIF_dia	58.52	42.03	83.88	61.47 $\pm$ 0.80
FOIF_hvp	56.81	40.36	83.39	60.19 $\pm$ 0.61
FOIF_ide	57.86	42.11	78.82	59.60 $\pm$ 2.10
FOIF_opt	57.43	40.86	81.65	59.98 $\pm$ 0.43
EMB	58.55	40.77	83.25	60.86 $\pm$ 0.14
Top_PPL	57.59	41.94	82.70	60.74 $\pm$ 0.65
Mid_PPL	58.59	41.11	81.57	60.42 $\pm$ 0.56
Random	58.72	41.72	81.22	60.56 $\pm$ 0.44
Orig	59.88	40.50	40.66	47.01

Table 17: Held-out test performance on Qwen3-1.7B (Strategy 2). The macro-average entry is reported as mean $\pm$ standard deviation over three independent runs.

SQuAD setting and does not imply that $\alpha = 0.01$ is universally optimal.

Regarding the number of perturbations K , the best observed performance generally occurs within the tested range of 10 to 30, although the trend varies across ZOIF variants. A larger direction budget reduces Monte Carlo error for the ZO estimator, while finite-radius bias and the structural Step 2 residual remain. The non-monotonic results therefore indicate diminishing and variant-dependent returns over the tested range. Theorem 2 further gives an effective-dimension interpretation when both the computation and perturbations are explicitly restricted to a low-rank subspace.

E.3.2 Selection Budget Sensitivity

We further investigate the impact of data selection budget on model performance by conducting a sensitivity analysis on Llama-3.2-1B (SQuAD, Strategy 2) across selection ratios of 2.5%, 5.0%, 10.0%, and 15.0%. As summarized in Table 19, the results reveal a non-monotonic trend: performance generally improves as the budget increases from 2.5% to 10.0%, but declines at the 15.0% ratio ($\sim 30k$ samples) for most ZOIF variants and the Random baseline. This observation suggests that data selection is not governed by a simple “more-is-better” principle; beyond an optimal threshold, expanding the selection budget may introduce redundancy or distributional shift, thereby diluting the quality of the curated subset and impairing fine-

α Sensitivity (fixed $K = 30$)			
ZOIF Variant	$\alpha = 0.001$	$\alpha = 0.01$	$\alpha = 0.1$
ZOIF_dia	62.40	69.36	65.19
ZOIF_hvp	59.28	69.43	64.08
ZOIF_ide	61.95	67.06	55.95
ZOIF_opt	65.22	70.58	61.02

K Sensitivity (fixed $\alpha = 0.001$)			
ZOIF Variant	$K = 5$	$K = 10$	$K = 30$
ZOIF_dia	60.22	61.63	62.40
ZOIF_hvp	44.41	65.98	59.28
ZOIF_ide	52.17	56.57	61.95
ZOIF_opt	61.19	67.31	65.22

Table 18: Performance sensitivity to perturbation radius α (fixed $K = 30$) and random directions K (fixed $\alpha = 0.001$) on Llama-3.2-1B (SQuAD).

Method	2.5%	5.0%	10.0%	15.0%
ZOIF_dia	60.66	62.40	73.18	69.69
ZOIF_hvp	61.06	59.28	58.61	60.66
ZOIF_ide	64.80	61.95	64.96	62.91
ZOIF_opt	62.50	65.22	69.08	65.39
Random	58.80	62.41	65.23	64.33

Table 19: Sensitivity analysis of selection budget on Llama-3.2-1B (SQuAD, Strategy 2).

tuning efficacy.

Across the majority of selection ratios, ZOIF_dia and ZOIF_opt consistently outperform random sampling, with peak gains observed at moderate budgets (5.0%–10.0%). Notably, ZOIF_ide achieves the highest score at the most constrained setting (2.5%), highlighting its robustness under extreme resource limitations. Collectively, these results corroborate ZOIF’s capacity to reliably identify high-influence samples and demonstrate its superior gain efficiency: under tight annotation or computational budgets, ZOIF delivers stronger performance per selected sample compared to naive baselines. The observed performance decay at 15.0% further underscores the importance of adaptive budget calibration in practical deployment scenarios.

E.4 Stability of Ranking Correlation (ZOIF vs. FOIF)

To assess the ranking stability of our approach, we conduct a comparative analysis between ZOIF and

Method	Top- K Overlap	Spearman $\rho_{\cap}$
ZOIF_dia	16.44%	0.3277
ZOIF_hvp	19.74%	0.2725
ZOIF_ide	27.41%	0.2871
ZOIF_opt	19.72%	0.4096

Table 20: Consistency analysis: ZOIF vs. FOIF ($K = 10,000$).

FOIF (used as the first-order reference) on Llama-3.2-1B using the SQuAD dataset. The evaluation employs two complementary metrics: (1) *Top- K Overlap*, which quantifies the agreement on the most influential samples, and (2) *Spearman Rank Correlation at Intersection* ($\rho_{\cap}$), which measures the monotonic relationship within the co-selected Top- K subset ($K = 10,000$). The quantitative results are presented in Table 20.

The results show measurable alignment between ZOIF and the reference FOIF rankings from both macroscopic and microscopic perspectives. Macroscopically, the Top- K overlap spans 16.44% to 27.41%, exceeding the expected random baseline of $\sim 5\%$ and indicating that ZOIF recovers part of the high-value sample regions identified by FOIF. Microscopically, the Spearman correlation peaks at 0.4096 for the ZOIF_opt variant, indicating moderate consistency in relative rankings despite the inherent noise of zeroth-order estimation. Together with the downstream results, this partial ranking agreement helps explain how ZOIF achieves performance comparable to FOIF while substantially reducing memory overhead.