ACE: LIFTING DATA ATTRIBUTION FROM SAMPLES TO CLUSTERS VIA INTRINSIC GRADIENT SIMILARITY

Anonymous authors
Paper under double-blind review

ABSTRACT

Data quality has become increasingly central to language model training, making effective data selection important. Yet many influential data-selection methods score each sample independently at high cost. Such pointwise scoring overlooks similarities in gradient behavior across samples, leading to limited scalability and redundant selection. To address these limitations, we introduce **ACE** (Data **A**ttribution via **C**luster-Level **E**stimation), a scalable framework that exploits intrinsic gradient similarity to lift both utility estimation and budget allocation from individual samples to clusters. ACE groups samples into clusters by gradient similarity, scores a few representatives per cluster to estimate cluster-level utility, and allocates the selection budget across clusters. We prove that the resulting estimator is unbiased and has controlled variance, providing a principled basis for reliable cluster-level attribution. Across post-training experiments, ACE achieves an end-to-end speedup of up to $\mathbf{33.8}\times$ and consistently outperforms its sample-level counterparts. Using only $\mathbf{10\%}$ of the data, it even surpasses **full-data** training on Llama-3.2-3B and Mistral-7B. Pre-training experiments further validate its scalability to larger data pools: ACE improves upon sample-level OPUS, achieves an $\mathbf{8.86}\times$ speedup, and delivers the strongest overall results among the baselines. Code is available in our `anonymous repository`.

1 INTRODUCTION

Large language models have demonstrated strong capabilities across a wide range of tasks (Brown et al., 2020; Chowdhery et al., 2022; Touvron et al., 2023). Yet as model architectures and optimization recipes become increasingly standardized, training data has become a key factor shaping model performance and generalization (Hoffmann et al., 2022; Kaplan et al., 2020; Longpre et al., 2024). This makes it increasingly important to identify high-quality data. Instead of relying only on global quality filters or corpus-level heuristics, **data attribution** (Deng et al., 2025) provides a principled way to estimate the utility of each candidate sample, thereby enabling data selection or reweighting.

Data attribution offers several ways to estimate the utility of training examples. Influence-function methods estimate how each training sample affects a target loss (Koh & Liang, 2017); trajectory-based methods such as TracIn accumulate gradient interactions across checkpoints (Pruthi et al., 2020); and gradient-similarity methods such as LESS measure the alignment between candidate and target gradients (Xia et al., 2024). After scoring each candidate, these methods usually select the top-ranked examples. Although they differ in their scoring objectives, they all treat each sample as an independent attribution unit and score it separately. This creates two limitations. **Limited scalability**: scoring every example independently makes attribution cost grow linearly with the candidate pool, which becomes expensive for large models and corpora. **Redundant selection**: many gradient-similar examples can simultaneously occupy the top of the ranking, concentrating the selected data in narrow regions of the target set. This redundancy arises because training corpora naturally contain groups of examples that are close in content, domain, or gradient. These intrinsic similarities should therefore be treated not merely as redundancy, but as a structure to exploit.

To address these limitations, we introduce **ACE** (Data **A**ttribution via **C**luster-Level **E**stimation), a scalable framework that treats clusters rather than individual samples as the units of attribution. ACE first constructs gradient-consistent clusters using projected gradients from a lightweight proxy model, whose induced data structure empirically transfers across model scales and architectures. It

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

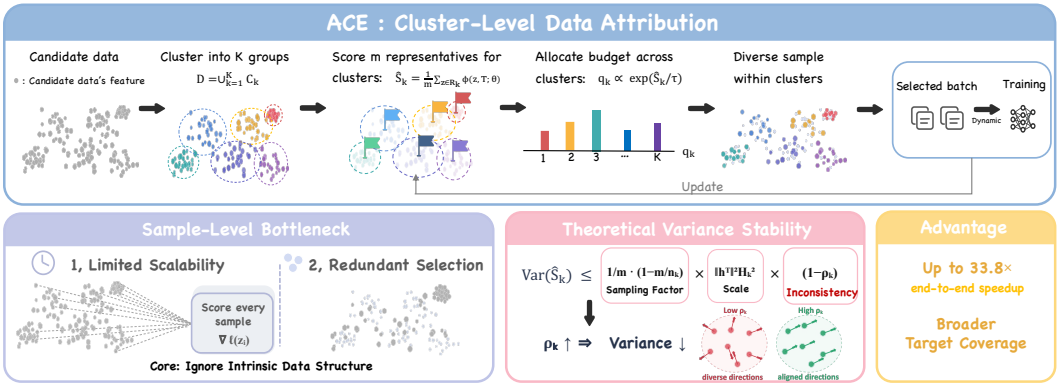


Figure 1. **Overview of ACE.** ACE estimates attribution scores at the cluster level using representative samples, and then uses the resulting scores for selection, sampling, or reweighting.

then applies the original attribution method to only a few representatives per cluster, aggregates their scores to estimate cluster utility, and allocates the selection budget across clusters. This reduces redundant per-sample evaluation while preventing similar examples from dominating the selected set, thereby improving coverage and diversity. However, clustering is not a trivial preprocessing step: the clusters must be aligned with the underlying attribution behavior for representative-based estimation to be reliable.

To make ACE reliable and practical, we study both its statistical properties and how to construct effective clusters. We prove that the representative-based cluster estimator is unbiased, with variance controlled by the number of representatives and the **intra-cluster gradient similarity**. With the number of representatives fixed, we evaluated a wide range of proxy features for cluster construction and identified small-model projected gradients as the most suitable choice for building transferable gradient-consistent clusters.

Guided by this analysis, we primarily evaluate ACE in post-training and further extend it to pre-training to assess its scalability on substantially larger candidate pools. In post-training, ACE substantially reduces gradient computation, delivers end-to-end speedups of up to $\mathbf{33.8}\times$, and consistently improves average performance over its sample-level counterparts. With only $\mathbf{10}\%$ of the data, it even outperforms **full-data** training on Llama-3.2-3B and Mistral-7B. Pre-training results further confirm that ACE remains effective at larger data scales, achieving the strongest overall performance among the evaluated data-selection baselines, including the $\mathbf{2}\times$token random baseline.

Our contributions are summarized as follows:

- We introduce **ACE**, a general framework that **exploits intrinsic data gradient similarity** to lift attribution, selection, and reweighting from samples to clusters, reducing redundant computation while promoting data coverage and diversity.
- We provide a comprehensive theoretical and empirical analysis of attribution clustering. We characterize the properties of effective clusters, prove that representative-based cluster attribution is **unbiased** with **variance controlled** by the number of representatives and intra-cluster gradient similarity, and show that **small-model projected gradients** produce gradient-consistent clusters that transfer across model scales and architectures.
- Across post-training and pre-training experiments, ACE achieves up to $\mathbf{33.8}\times$ end-to-end speedup and consistently outperforms its sample-level counterparts on average under the same training budget. On Llama-3.2-3B and Mistral-7B, ACE further outperforms **full-data** training while using only $\mathbf{10}\%$ of the data, and achieves the strongest results among the evaluated baselines.

2 Related Work

Data Selection and Reweighting Data selection improves training efficiency by identifying or up-weighting useful examples under a fixed budget. Sample-level methods estimate the utility of individual examples using influence functions (Koh & Liang, 2017; Grosse et al., 2023; Kwon et al.,

2024), gradient-based attribution (Park et al., 2023; Xia et al., 2024), or training trajectories and dynamics (Pruthi et al., 2020; Wang et al., 2026; Gu et al., 2025; Zhao et al., 2026). Although effective, these methods often require per-example features, repeated scoring, or checkpoint-wise computation, which limits their scalability to large models and corpora. Another line of work optimizes sampling weights over domains, skills, or data groups (Xie et al., 2023; Chen et al., 2023). These methods are more scalable, but often assume that meaningful groups or metadata are available in advance. Our framework avoids this requirement by automatically discovering gradient-consistent clusters and using them to lift existing sample-level attribution rules to the cluster level.

Clustering for Data Selection. Clustering has been widely used to reduce redundancy and construct representative subsets, including semantic deduplication methods (Lee et al., 2022; Abbas et al., 2023). Recent methods use clusters for different purposes: ClusterClip rebalances semantic-cluster frequencies (Shao et al., 2024); TAGCOS performs task-agnostic coreset selection over full-pool gradient representations (Zhang et al., 2025); ClusterUCB allocates sample-level influence evaluations using a bandit objective (Wang et al., 2025); and Clustered Influence Functions amortizes repeated subset queries through cluster-mean caching (Badó & Fenech, 2026). ACE takes a different route: we first show that attribution-relevant learning signals exhibit stable and transferable cluster-level structure, and then exploit this structure for data selection. ACE estimates the target-conditioned utility of each cluster from only a few representatives, avoiding full-pool attribution while providing an unbiased estimator with controlled variance.

3 METHOD

3.1 PRELIMINARIES: SAMPLE-LEVEL DATA ATTRIBUTION

Let $\mathcal{D} = \{z_i\}_{i=1}^N$ be a candidate dataset and $\mathcal{T}$ a target set. For each candidate example, we compute an attribution feature

$$h_i = h(z_i;\theta) \in \mathbb{R}^d, \qquad h_{\mathcal{T}} = \frac{1}{|\mathcal{T}|}\sum_{z\in\mathcal{T}} h(z;\theta). \tag{1}$$

Depending on the underlying method, h may represent gradients, projected gradients, optimizer-preconditioned gradients, or checkpoint-wise gradient features.

A broad class of gradient-based attribution methods assigns each example the score

$$s_i = \langle h_i, h_{\mathcal{T}} \rangle, \tag{2}$$

where method-specific normalization is absorbed into h. For a selection budget $B \ll N$, let Top_B denote the B candidates with the largest scores s_i. The scores can be used either for hard selection or for softmax reweighting:

$$\mathcal{S}_B = \mathrm{Top}_B\{z_i\}_{i=1}^N \quad \text{or} \quad p_i = \frac{\exp(s_i/\tau)}{\sum_{j=1}^N \exp(s_j/\tau)}, \qquad \tau > 0. \tag{3}$$

As model and corpus sizes grow, computing and storing attribution features for all N candidates, together with the associated forward–backward and scoring costs, becomes prohibitive.

3.2 CLUSTER-LEVEL DATA ATTRIBUTION

Clustering the candidate corpus. We replace individual examples with clusters as the atomic units of attribution. Let K denote the number of clusters. For each example, we extract a clustering feature $r_i = \psi(z_i) \in \mathbb{R}^{d_r}$ and partition the corpus using standard K-means:

$$\{C_k\}_{k=1}^K = \mathrm{KMeans}\big(\{r_i\}_{i=1}^N, K\big), \qquad \mathcal{D} = \bigcup_{k=1}^K C_k, \qquad C_k \cap C_\ell = \varnothing \text{ for } k \neq \ell. \tag{4}$$

Here, r_i is a low-cost proxy-model feature used for clustering, whereas h_i is the method-specific feature used for attribution. We deliberately adopt a simple clustering algorithm and focus on finding an inexpensive feature ψ that groups examples with consistent attribution features. Further analysis of the clustering properties is provided in Section 4.2.

Table 1. Complexity and ideal attribution-stage speedup.

Method	Gradient Evaluations	Total Cost	Speedup
Sample-Level	N	$O(NC_g)$	$1\times$
Cluster-Level	Km	$O(NC_\psi + INKd_r + KmC_g)$	$\approx \frac{N}{Km}\times$

Representative-based attribution. Let $n_k = |C_k|$. The exact utility of cluster C_k is the average utility of its members:

$$S_k = \frac{1}{n_k}\sum_{z_i \in C_k} s_i = \left\langle \bar{h}_k, h_{\mathcal{T}} \right\rangle, \qquad \bar{h}_k = \frac{1}{n_k}\sum_{z_i \in C_k} h_i. \tag{5}$$

Computing $\bar{h}_k$ exactly would still require scoring every example. We therefore sample m representatives $R_k \subseteq C_k$ uniformly without replacement and estimate

$$\widehat{h}_k = \frac{1}{m}\sum_{z_i \in R_k} h_i, \qquad \widehat{S}_k = \left\langle \widehat{h}_k, h_{\mathcal{T}} \right\rangle = \frac{1}{m}\sum_{z_i \in R_k} s_i. \tag{6}$$

Cluster-level budget allocation. The estimated utilities define a cluster-level sampling distribution $q = (q_1, \ldots, q_K)$, where q_k is the probability assigned to cluster C_k. We obtain this distribution by solving the entropy-regularized utility maximization problem

$$q^\star = \arg\max_{q \in \Delta_K} \left\{ \sum_{k=1}^{K} q_k \widehat{S}_k + \tau \mathcal{H}(q) \right\}, \qquad \mathcal{H}(q) = -\sum_{k=1}^{K} q_k \log q_k, \tag{7}$$

where $\Delta_K = \{q \in \mathbb{R}_+^K : \sum_{k=1}^K q_k = 1\}$. The first-order condition, with λ denoting the multiplier for the normalization constraint, gives

$$\widehat{S}_k - \tau(1 + \log q_k^\star) - \lambda = 0 \quad \Longrightarrow \quad q_k^\star = \frac{\exp(\widehat{S}_k/\tau)}{\sum_{\ell=1}^K \exp(\widehat{S}_\ell/\tau)}. \tag{8}$$

Under a total budget B, sampling a cluster according to $q^\star$ and then sampling uniformly within it gives, for $z_i \in C_k$, the per-draw probability and expected cluster allocation

$$p_i^{\mathrm{ACE}} = \frac{q_k^\star}{n_k}, \qquad \mathbb{E}[B_k] = B q_k^\star, \tag{9}$$

where B_k is the number of selected examples from C_k. The temperature τ controls the trade-off between high-utility clusters and broader coverage.

ACE thus preserves the underlying attribution score while lifting both utility estimation and budget allocation to clusters. Detailed instantiations are provided in Appendix A.

Why do a few representatives suffice?

Definition 1 (Intra-Cluster Attribution Similarity)**.** *Let $\{C_k\}_{k=1}^K$ be a partition of the candidate dataset, where $n_k = |C_k|$, and let h_i denote the attribution feature of z_i. We define the cluster-level and mean intra-cluster similarities as*

$$\rho_k = \frac{1}{n_k(n_k-1)} \sum_{\substack{z_i, z_j \in C_k \\ i \neq j}} \frac{\langle h_i, h_j \rangle}{\|h_i\|\,\|h_j\|}, \qquad \bar{\rho} = \frac{1}{K}\sum_{k=1}^{K} \rho_k. \tag{10}$$

A larger ρ_k indicates stronger alignment among the attribution directions within C_k, whereas a larger $\bar{\rho}$ indicates greater overall consistency across the clustering.

Theorem 2 (Variance of Cluster-Level Attribution)**.** *Suppose that R_k contains m examples sampled uniformly without replacement from C_k, and that $\|h_i\| = H_k$ for all $z_i \in C_k$. Then the representative-based estimator is unbiased:*

$$\mathbb{E}[\widehat{S}_k] = S_k. \tag{11}$$

Moreover, its variance satisfies

$$\mathrm{Var}(\widehat{S}_k) \leq \frac{1}{m}\left(1 - \frac{m}{n_k}\right) \|h_{\mathcal{T}}\|^2 H_k^2 (1 - \rho_k). \tag{12}$$

Theorem 2 first establishes the reliability of representative-based attribution within each cluster. Let $H_{\max} := \max_{1 \le k \le K} H_k$. When $n_k \gg m$, the per-cluster bound simplifies, while averaging the original bound over all clusters gives the corresponding dataset-level guarantee:

$$\mathrm{Var}(\widehat{S}_k) \lesssim \frac{\|h_{\mathcal{T}}\|^2 H_k^2}{m}(1-\rho_k), \qquad \mathcal{V}_{\mathrm{all}} := \frac{1}{K}\sum_{k=1}^{K} \mathrm{Var}(\widehat{S}_k) \le \frac{\|h_{\mathcal{T}}\|^2 H_{\max}^2}{m}(1-\bar{\rho}). \tag{13}$$

The first bound shows that few representatives suffice when cluster members have consistent gradient directions, while the second shows that dataset-level estimation variance is controlled by the mean intra-cluster similarity $\bar{\rho}$. Thus, $\bar{\rho}$ provides a principled criterion for cluster construction and motivates our analysis in Section 4.2.

Complexity. Let C_ψ and C_g denote the per-example costs of extracting clustering and attribution features, respectively. Let I denote the number of K-means iterations. Cluster-level attribution reduces expensive gradient evaluations from N to Km. When $Km \ll N$ and the clustering feature is inexpensive, its attribution stage is accelerated by approximately $N/(Km)$.

4 Experiments

4.1 Experimental Setup

We evaluate ACE in both post-training and pre-training settings. Within each setting, all methods share the same backbone, candidate pool, selected-data budget, and training pipeline; only the data-selection policy is changed. Additional experiment details are provided in Appendix C.

Post-training setup. We conduct post-training experiments using DataFlex (Liang et al., 2026). Following LESS (Xia et al., 2024), we fine-tune Llama-3.2-3B, Llama-2-7B, Mistral-7B using LoRA (Hu et al., 2022) with rank 128. We train for one epoch with an effective batch size of 64, sequence length of 2,048, learning rate 2×10^{-5}. The candidate pool contains 270,679 instruction-tuning examples from FLAN-v2, Chain-of-Thought, Dolly, and OpenAssistant-1. The target set is *LESS_eval*, consisting of 375 examples from MMLU, BBH, and TyDiQA. Dynamic selection starts after 20 warm-up steps and runs every 100 optimizer steps for four rounds. Each round selects 6,400 examples, resulting in a total budget of 25,600 examples, approximately 10% of the candidate pool.

Pre-training setup. To further evaluate ACE's scalability to substantially larger data volumes, we extend our experiments to pre-training. We conduct pre-training experiments using TorchTitan (Liang et al., 2025). Following OPUS (Wang et al., 2026), we pre-train a GPT-2 Large model from scratch using a sequence length of 2,048 on FineWeb. We train for 30B tokens, corresponding to 57,220 steps, with a global batch size of 524,288 tokens on 32 H20 GPUs. The target set is a proxy containing 29,626 sequences of length 512, drawn from the downstream evaluation distribution.

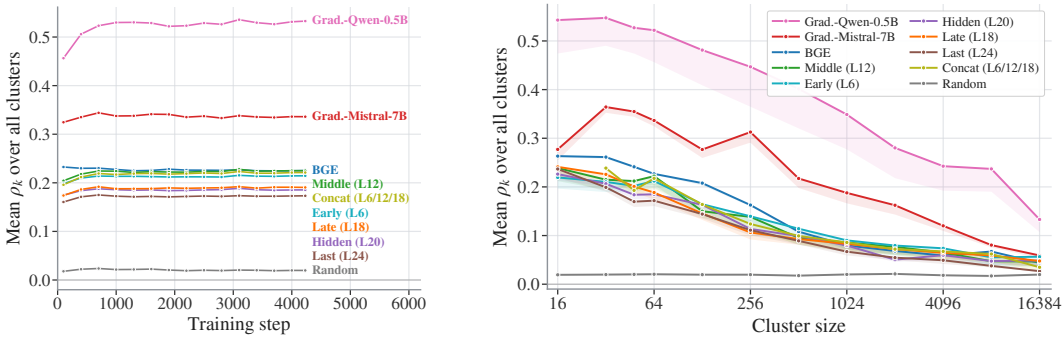
Clustering details. We construct clusters once before selection using projected gradient features from Qwen2.5-0.5B. We use a target cluster size of 64 for post-training and $K = 1{,}024$ clusters for pre-training on FineWeb. Cluster utilities are estimated from $m = 8$ uniformly sampled representatives, with cached features and assignments reused across rounds and attribution objectives.

Baselines. In post-training, we compare against full-data training, random sampling, embedding-based selectors NEAR and TSDS (Liu et al., 2024), online reweighting with ADAPT (Zhao et al., 2026), and sample-level LESS (Xia et al., 2024), TracIn (Pruthi et al., 2020), and loss-based selection. In pre-training, we compare against random sampling, a 2×token random run, static quality filters FineWeb-Edu (Penedo et al., 2024), DCLM (Li et al., 2024), and Qurating (Wettig et al., 2024), and sample-level GREATs (Wang et al., 2024) and OPUS (Wang et al., 2026).

Evaluation. For post-training, we use **lm-evaluation-harness** (Gao et al., 2024) following the LESS protocol: 5-shot accuracy on MMLU (Hendrycks et al., 2021), 3-shot exact match on BBH (Suzgun et al., 2023), and 1-shot on TyDiQA (Clark et al., 2020). For pre-training, we follow OPUS (Wang et al., 2026) and evaluate zero-shot log-likelihood performance on MMLU (Hendrycks et al., 2021), ANLI (Nie et al., 2020), HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), SIQA (Sap et al., 2019), WinoGrande (Sakaguchi et al., 2020), ARC-Challenge (Clark et al., 2018), CommonsenseQA (Talmor et al., 2019), WSC (Levesque et al., 2012), and RACE (Lai et al., 2017). We report unweighted task averages, with post-training results averaged over three runs. Efficiency is measured by cold-start end-to-end runtime, including embedding, clustering, selection, and training.

4.2 Does Training Data Exhibit Cluster-Level Gradient Structure?

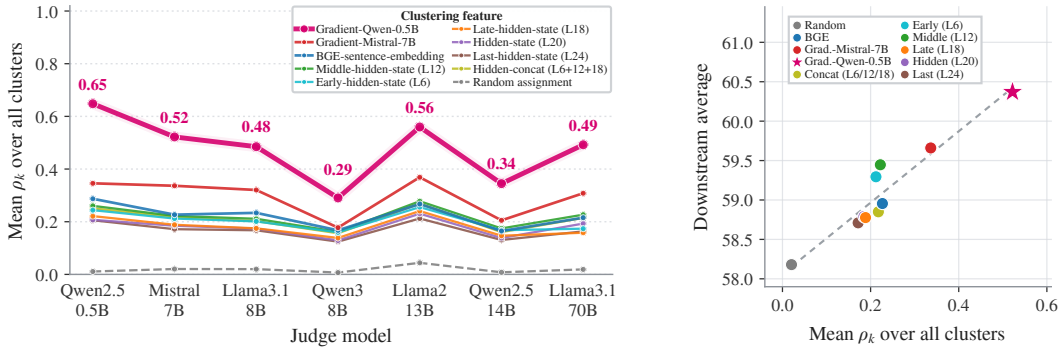
Sample-level attribution treats every example as an independent unit. Yet many examples may push the model in similar gradient directions. ACE **exploits this shared structure** by using a few representatives to estimate the utility of an entire cluster. Theorem 2 shows that such estimation becomes more reliable as gradients within a cluster become more aligned. We now examine whether this structure is **stable throughout training**, **transfers across models**, and **tracks selection quality**. Using the 270K-example post-training pool, we measure cluster quality by the mean within-cluster gradient similarity $\bar{\rho}$. We compare clusters built from gradients, semantic embeddings, hidden states, and random assignment, using separate Mistral-7B gradients computed throughout training to assess whether each partition captures coherent gradient structure. Details are provided in Appendix E.



(a) $\bar{\rho}$ across training checkpoints. *(b)* $\bar{\rho}$ under different cluster sizes.

Figure 2. Analysis of gradient-consistent clustering. **Left:** Small-model gradient features maintain high intra-cluster similarity throughout the entire training process. **Right:** This advantage remains stable across cluster sizes. Solid curves show checkpoint means; shaded regions show minimum values.

The structure persists throughout training and across practical granularities. Figure 2a shows that projected gradients from Qwen2.5-0.5B consistently produce the most coherent clusters across checkpoints. At the default cluster size of 64, they achieve $\bar{\rho} = 0.522$, compared with 0.021 for random assignment. The stable gap throughout training indicates that the observed coherence is not tied to a particular checkpoint. Figure 2b further varies the cluster size from 16 to 16,384, using eight representatives per cluster. For Qwen2.5-0.5B gradients, $\bar{\rho}$ increases slightly from 0.543 at size 16 to 0.547 at size 32, and then gradually decreases as clusters become coarser. This reveals an important boundary of the structure: nearby examples share coherent learning signals, but excessive aggregation eventually merges different gradient directions. We therefore use size 64, which retains near-peak coherence while substantially reducing the number of attribution units.



(a) $\bar{\rho}$ across seven training models. *(b)* $\bar{\rho}$ vs. downstream performance.

Figure 3. Analysis of gradient-consistent clustering. **Left:** Clustering-feature rankings remain consistent across models of different sizes and architectures. **Right:** Higher intra-cluster gradient consistency correlates with better downstream data-selection performance.

The structure transfers across models and tracks selection quality. Figure 3a evaluates the same clusters using seven gradient evaluators ranging from Qwen2.5-0.5B to Llama-3.1-70B. Clusters constructed from Qwen2.5-0.5B gradients achieve the highest similarity under every evaluator, while the ranking of clustering representations remains highly stable across models, with Spearman

Table 2. **Post-training results across three backbones.** Full Data uses 270K examples, while all selection methods use 25.6K (approximately 10%). Scores are averaged over three runs, with sample standard deviations shown as gray subscripts. Δ Avg. and Speedup are relative to the corresponding sample-level variant; Time includes embedding, clustering, selection, and training. Within each backbone, the best 10% score in each performance column is bold, while the cluster-level variant with the highest Avg. is highlighted in darker blue. Corresponding results on four diverse OOD benchmarks are reported in Appendix D.

Model	Method	Variant	Data ratio	Downstream performance				Efficiency		
				MMLU	BBH	TyDiQA	Avg.↑	Δ Avg.	Time↓	Speedup↑
Llama-3.2-3B	Random	Uniform Sampling	10%	$55.63_{\pm 0.19}$	$45.56_{\pm 0.37}$	$61.28_{\pm 1.03}$	$54.16_{\pm 0.26}$	–	–	–
	TSDS	Embedding-Based	10%	$55.09_{\pm 0.10}$	$45.90_{\pm 0.14}$	$63.18_{\pm 0.31}$	$54.72_{\pm 0.11}$	–	–	–
	ADAPT	Online Reweighting	10%	$54.95_{\pm 0.70}$	$44.78_{\pm 1.81}$	$62.43_{\pm 2.68}$	$54.06_{\pm 1.63}$	–	–	–
	NEAR	Embedding-Based	10%	$55.27_{\pm 0.31}$	$45.00_{\pm 0.67}$	$62.84_{\pm 0.57}$	$54.37_{\pm 0.45}$	–	–	–
	TracIn	Sample-Level	10%	$55.62_{\pm 0.09}$	$45.22_{\pm 0.96}$	$61.89_{\pm 1.07}$	$54.24_{\pm 0.52}$	–	29h20m	1.0×
	TracIn	**Cluster-Level**	10%	$55.66_{\pm 0.44}$	$45.62_{\pm 0.99}$	$63.98_{\pm 0.40}$	$55.09_{\pm 0.38}$	**+0.85**	52m	**33.8×**
	Loss	Sample-Level	10%	$55.52_{\pm 0.27}$	$45.43_{\pm 0.23}$	$61.29_{\pm 0.06}$	$54.08_{\pm 0.16}$	–	4h27m	1.0×
	Loss	**Cluster-Level**	10%	$\mathbf{55.96}_{\pm 0.24}$	$45.19_{\pm 1.46}$	$\mathbf{64.75}_{\pm 1.07}$	$\mathbf{55.30}_{\pm 0.18}$	**+1.22**	47m	**5.7×**
	LESS	Sample-Level	10%	$55.38_{\pm 0.47}$	$46.05_{\pm 0.42}$	$60.73_{\pm 1.36}$	$54.05_{\pm 0.29}$	–	23h23m	1.0×
	LESS	**Cluster-Level**	10%	$55.66_{\pm 0.47}$	$\mathbf{46.60}_{\pm 0.42}$	$62.77_{\pm 0.77}$	$55.01_{\pm 0.18}$	**+0.96**	53m	**26.5×**
	Full Data	All Examples	100%	$54.00_{\pm 0.10}$	$44.75_{\pm 0.42}$	$65.55_{\pm 0.35}$	$54.77_{\pm 0.17}$	–	3h26m	–
Llama-2-7B	Random	Uniform Sampling	10%	$46.37_{\pm 0.41}$	$38.73_{\pm 0.35}$	$52.51_{\pm 0.21}$	$45.87_{\pm 0.31}$	–	–	–
	TSDS	Embedding-Based	10%	$45.54_{\pm 0.42}$	$39.38_{\pm 0.53}$	$\mathbf{54.82}_{\pm 0.82}$	$46.58_{\pm 0.42}$	–	–	–
	ADAPT	Online Reweighting	10%	$45.96_{\pm 0.93}$	$39.41_{\pm 1.26}$	$54.33_{\pm 0.65}$	$46.57_{\pm 0.14}$	–	–	–
	NEAR	Embedding-Based	10%	$46.26_{\pm 0.07}$	$37.50_{\pm 0.85}$	$53.38_{\pm 0.48}$	$45.71_{\pm 0.46}$	–	–	–
	TracIn	Sample-Level	10%	$46.12_{\pm 0.43}$	$37.44_{\pm 0.23}$	$54.32_{\pm 0.79}$	$45.96_{\pm 0.41}$	–	36h51m	1.0×
	TracIn	**Cluster-Level**	10%	$\mathbf{47.33}_{\pm 0.15}$	$\mathbf{40.43}_{\pm 0.84}$	$54.10_{\pm 0.62}$	$\mathbf{47.29}_{\pm 0.08}$	**+1.33**	1h15m	**29.5×**
	Loss	Sample-Level	10%	$46.01_{\pm 0.21}$	$39.10_{\pm 0.51}$	$52.98_{\pm 0.24}$	$46.03_{\pm 0.30}$	–	4h01m	1.0×
	Loss	**Cluster-Level**	10%	$46.96_{\pm 0.15}$	$39.66_{\pm 0.14}$	$53.25_{\pm 0.13}$	$46.62_{\pm 0.03}$	**+0.59**	1h10m	**3.4×**
	LESS	Sample-Level	10%	$45.70_{\pm 0.98}$	$38.30_{\pm 1.07}$	$54.53_{\pm 0.47}$	$46.18_{\pm 0.58}$	–	34h27m	1.0×
	LESS	**Cluster-Level**	10%	$46.94_{\pm 0.47}$	$39.85_{\pm 0.19}$	$53.68_{\pm 0.51}$	$46.82_{\pm 0.11}$	**+0.64**	1h16m	**27.2×**
	Full Data	All Examples	100%	$47.41_{\pm 0.17}$	$40.52_{\pm 0.62}$	$56.96_{\pm 0.37}$	$48.30_{\pm 0.30}$	–	8h01m	–
Mistral-7B	Random	Uniform Sampling	10%	$60.18_{\pm 0.14}$	$56.08_{\pm 0.74}$	$58.43_{\pm 0.69}$	$58.23_{\pm 0.03}$	–	–	–
	TSDS	Embedding-Based	10%	$60.58_{\pm 0.20}$	$55.22_{\pm 0.84}$	$60.47_{\pm 0.67}$	$58.76_{\pm 0.49}$	–	–	–
	ADAPT	Online Reweighting	10%	$59.83_{\pm 1.28}$	$54.66_{\pm 0.95}$	$\mathbf{61.29}_{\pm 1.29}$	$58.59_{\pm 0.53}$	–	–	–
	NEAR	Embedding-Based	10%	$59.08_{\pm 0.29}$	$51.88_{\pm 2.13}$	$58.11_{\pm 2.19}$	$56.36_{\pm 0.79}$	–	–	–
	TracIn	Sample-Level	10%	$58.45_{\pm 0.42}$	$49.01_{\pm 2.83}$	$56.28_{\pm 1.04}$	$54.58_{\pm 1.28}$	–	38h07m	1.0×
	TracIn	**Cluster-Level**	10%	$60.72_{\pm 0.54}$	$55.80_{\pm 1.11}$	$60.06_{\pm 0.77}$	$58.86_{\pm 0.70}$	**+4.28**	1h45m	**21.8×**
	Loss	Sample-Level	10%	$59.89_{\pm 0.17}$	$56.73_{\pm 0.74}$	$58.44_{\pm 0.78}$	$58.35_{\pm 0.26}$	–	2h59m	1.0×
	Loss	**Cluster-Level**	10%	$60.61_{\pm 0.42}$	$56.73_{\pm 1.61}$	$59.79_{\pm 0.76}$	$59.04_{\pm 0.55}$	**+0.69**	1h09m	**2.6×**
	LESS	Sample-Level	10%	$\mathbf{61.62}_{\pm 0.79}$	$55.83_{\pm 1.61}$	$58.31_{\pm 0.82}$	$58.59_{\pm 0.26}$	–	26h52m	1.0×
	LESS	**Cluster-Level**	10%	$61.16_{\pm 0.14}$	$\mathbf{57.69}_{\pm 1.03}$	$60.26_{\pm 0.69}$	$\mathbf{59.70}_{\pm 0.58}$	**+1.11**	1h05m	**24.8×**
	Full Data	All Examples	100%	$60.39_{\pm 0.34}$	$55.00_{\pm 2.03}$	$62.07_{\pm 0.51}$	$59.15_{\pm 0.73}$	–	6h41m	–

correlations of 0.90–1.00. This suggests that the proxy gradients capture structure shared across models, rather than geometry specific to the proxy itself. More importantly, this structural metric is closely related to selection quality. Across ten clustering representations, Figure 3b shows a strong correlation between $\bar{\rho}$ and the average downstream score with Pearson $= \mathbf{0.949}$ and Spearman $= \mathbf{0.927}$. Qwen2.5-0.5B gradient clustering achieves both the highest similarity 0.522 and the best downstream average 60.37, whereas random clustering yields the lowest values 0.021 and 58.18.

Together, these results show that cluster-level coherence is persistent, transferable, and practically meaningful. Intra-cluster similarity is therefore not only a theoretical quantity controlling estimator variance, but also a useful criterion for constructing clusters for data selection. We next evaluate whether exploiting this structure improves attribution efficiency and downstream performance.

4.3 Post-Training Data Selection

Table 2 evaluates ACE with TracIn, Loss, and LESS on Llama-3.2-3B, Llama-2-7B, and Mistral-7B. All selection methods use 25.6K examples, approximately 10% of the 270K-example candidate pool. Each cluster-level variant is compared with its sample-level counterpart under the same scoring objective, backbone, and training budget. We report performance on MMLU, BBH, and TyDiQA and their average, with means and sample standard deviations over three runs on 8 H20 GPUs.

Cluster-level estimation enables large end-to-end speedups. By estimating cluster utilities from a few representatives, ACE achieves **21.8–33.8**× speedups for TracIn, **24.8–27.2**× for LESS, and 2.6–5.7× for Loss across the three backbones. These end-to-end timings include embedding, clustering,

Table 3. **Pre-training results on GPT-2-large.** All methods use 30B training tokens except Random-60B. GPU Hours reports the total cold-start end-to-end cost, including feature extraction, clustering where applicable, attribution scoring, data selection, and training. The best 30B-token result per benchmark is bold. Blue and gray shading denote Cluster-OPUS and Random-60B, respectively; darker blue marks the best average.

Task	Data-selection baselines					OPUS family		Random-60B
	Random	FineWeb-Edu	DCLM	Qurating	GREATs	OPUS	Cluster-OPUS	
MMLU	23.89	**25.34**	23.12	23.51	25.31	24.88	24.73	24.44
ANLI	33.35	33.76	32.36	33.84	33.62	33.92	**34.51**	33.19
HellaSwag	55.34	55.11	54.73	**56.53**	54.48	54.81	55.54	57.26
PIQA	72.96	72.69	72.36	**73.23**	71.87	72.54	72.91	73.07
SIQA	**41.86**	40.96	41.50	40.99	39.71	40.84	40.99	41.15
W.G.	56.20	56.83	**58.01**	56.35	56.67	57.98	57.54	58.80
ARC-C	28.24	**29.69**	28.24	29.10	28.16	27.50	27.56	28.16
C.QA	18.51	19.96	19.74	**20.97**	19.41	20.12	20.64	19.66
WSC	61.17	66.42	**70.70**	67.40	67.40	69.84	69.23	68.13
RACE	34.64	34.77	33.49	34.16	34.07	34.58	**35.31**	34.64
Avg. ↑	42.62	43.55	43.43	43.61	43.07	43.70	**43.90**	43.85
GPU Hours ↓	–	–	–	–	–	6,912 h	**780 h (8.86×)**	–

selection, and training. On Llama-3.2-3B, Cluster-TracIn reduces runtime from 29h20m to 52m, yielding a **33.8**× speedup. Reusing cached features and cluster assignments further reduces the estimated per-run time to **28m**. Relative to the same reported sample-level runtime, this corresponds to approximately **61.3**× speedup for runs with cached preprocessing.

Surprisingly, ACE consistently improves average performance. Despite scoring far fewer candidate examples, ACE outperforms its sample-level counterparts in all nine selector–backbone comparisons under the same data budget. Gains in average downstream score range from **+0.59 to +4.28** points. The largest improvement occurs for TracIn on Mistral-7B, where the average score increases from 54.58 to 58.86 while the method runs 21.8× faster end to end. These gains extend across all three scoring criteria. ACE improves all nine comparisons across diverse **OOD** tasks in Appendix D.

Strong results against selection and full-data baselines. An ACE variant achieves the highest average score among the evaluated 10%-budget selection methods on every backbone: Cluster-Loss reaches **55.30** on Llama-3.2-3B, Cluster-TracIn reaches **47.29** on Llama-2-7B, and Cluster-LESS reaches **59.70** on Mistral-7B. Moreover, Cluster-Loss on Llama-3.2-3B exceeds full-data training by **0.53** points, while Cluster-LESS on Mistral-7B exceeds full-data training by **0.55** points, using only approximately 10% of the data. We later investigate the source of these gains in Section 4.5.

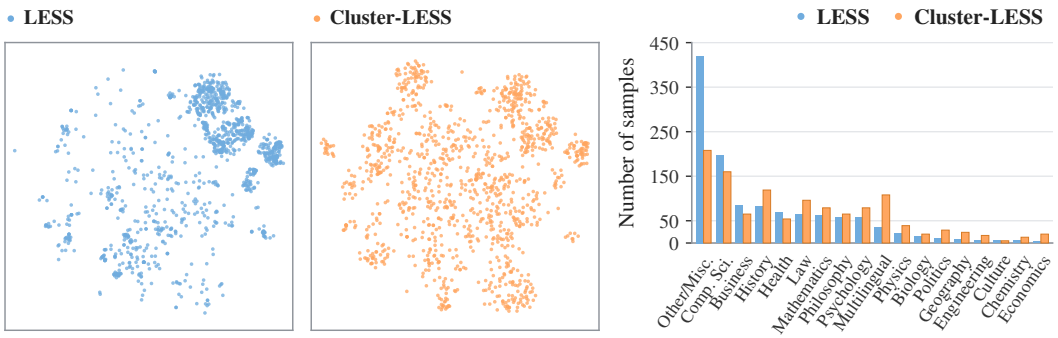
4.4 Pre-Training Data Selection

Table 3 evaluates whether ACE scales to pre-training data selection over substantially larger data volumes. We apply ACE to OPUS while keeping the selection objective, model architecture, and 30B-token training budget unchanged. OPUS performs attribution over individual samples, whereas Cluster-OPUS operates over gradient-consistent clusters.

Cluster-level attribution substantially reduces end-to-end training cost. Including feature extraction, clustering, attribution scoring, data selection, and training, Cluster-OPUS reduces the total cost from 6,912 to 780 GPU hours. It therefore requires only 11.3% of the OPUS cost, corresponding to an **8.86**× **reduction**. This comparison includes all one-time preprocessing and subsequent selection operations, rather than treating representation construction or clustering as free.

Cluster-OPUS requires 780.00 GPU hours in total, including 24.98 GPU hours for one-time gradient-representation extraction and clustering and **755.02** GPU hours for cluster-level attribution, selection, and training. Once computed, the representations and cluster assignments can be cached and reused in subsequent runs. Under this reuse setting, each additional run requires approximately 755.02 GPU hours, corresponding to a **9.15**× **speedup** over OPUS. The 8.86× speedup includes the one-time preprocessing cost and therefore represents the complete cold-start pipeline.

ACE maintains balanced performance across tasks. Under the same 30B-token budget, Cluster-OPUS improves the average downstream performance over OPUS from 43.70 to 43.90. More im-



(a) Coverage in the shared BGE–t-SNE space. *(b)* Coverage over target-set-derived domains.

Figure 4. Cluster-level selection improves target-relevant coverage. **Left:** Under the same selection budget, Cluster-LESS spans a broader region of the shared semantic space. **Right:** It yields a more balanced distribution across target-set-derived domains, avoiding narrow-domain concentration.

portantly, it exhibits a **balanced performance** profile across tasks, outperforming OPUS on seven of the ten benchmarks while showing only modest decreases on MMLU, WinoGrande, and WSC. Cluster-OPUS achieves the highest average among all 30B-token methods, surpassing FineWeb-Edu, Qurating, and DCLM. It also slightly outperforms Random-60B (43.85) while using only half as many training tokens. This balanced task-level performance motivates our subsequent analysis of whether cluster-level selection provides broader and more diverse data coverage.

4.5 Cluster-Level Selection Enhances Diversity

We use LESS as a representative sample-level selector to examine how cluster-level allocation changes the composition of selected data. Although LESS identifies individually high-scoring examples, pointwise ranking can repeatedly allocate the selection budget to examples aligned with a few dominant target directions. This results in redundant selections and leaves other target-relevant regions underrepresented.

Figure 4 compares LESS and Cluster-LESS using the same candidate pool and selection budget, with 1,200 selected examples visualized for each method. In the shared BGE–t-SNE space shown in the left panel, Cluster-LESS preserves target alignment while covering a broader and more evenly distributed region. Specifically, t-SNE coverage increases from 70.4% to 77.1% (**+6.7** pp), while BGE-cluster diversity increases from 27.3% to 37.2% (**+9.9** pp).

We further examine whether this broader coverage extends across target-relevant domains. As shown in the right panel, Cluster-LESS distributes selected examples more evenly across target-set-derived domains, whereas LESS concentrates on a small number of salient domains. Together, these results suggest that cluster-level allocation **reduces selection concentration** without sacrificing target relevance, providing a plausible explanation for the performance gains in Table 2. Details of the visualization protocol, coverage metrics, and target-domain construction are provided in Appendix F.

5 Conclusion

We introduced **ACE**, a scalable framework that exploits intrinsic gradient similarity to lift data attribution and selection **from individual samples to clusters**. By constructing gradient-consistent clusters and estimating their utilities from a few representatives, ACE reduces redundant computation while improving target-relevant coverage and selection diversity. We further prove that the resulting estimator is unbiased with controlled variance, providing a principled basis for reliable cluster-level attribution. In post-training, ACE achieves end-to-end speedups of up to $\mathbf{33.8}\times$, consistently outperforms its sample-level counterparts, and **surpasses full-data** training on Llama-3.2-3B and Mistral-7B using only $\mathbf{10}\%$ of the data. In pre-training, ACE improves upon sample-level OPUS with an $\mathbf{8.86}\times$ speedup and achieves the strongest overall results among the evaluated baselines, including the $\mathbf{2}\times$token random baseline. Together, these results establish cluster-level estimation as an efficient and effective alternative to exhaustive sample-level attribution, providing a practical approach to scaling data attribution and selection to larger data pools and models.

AI USE STATEMENT

Generative AI tools were used only for language editing and auxiliary category annotation. Language editing was limited to correcting grammar, refining sentence structure, and improving clarity after the scientific content had been developed by the authors.

GPT-6 was used as an external annotator for the diversity analysis in Figure 4(b). Given a fixed prompt and a predefined taxonomy, it assigned each displayed example to one category without access to the corresponding selection method. All annotations were manually reviewed by the authors for consistency. They were used solely for descriptive analysis and visualization, and did not affect data selection, model training, attribution scoring, or downstream evaluation.

Generative AI tools were not used to formulate the research problem, design ACE, develop the theoretical results, conduct the core experiments, or determine the conclusions. The authors reviewed all AI-assisted content and take full responsibility for the manuscript.

ETHICS STATEMENT

This work adheres to the ICLR Code of Ethics. It involves no human-subject or animal experiments and introduces no new collection or annotation of personal data. All datasets, evaluation benchmarks, and pretrained models are obtained from publicly released sources and used in accordance with their respective licenses and terms of use. Our study neither requires identifying individuals nor attempts to infer sensitive personal attributes.

As with other studies based on web and instruction-tuning corpora, the source datasets may contain social biases, harmful content, or residual personal information. Moreover, ACE performs target-conditioned data attribution, so its outputs necessarily depend on the composition of the candidate corpus and target set. An unrepresentative target set may favor particular domains or data distributions. Although cluster-level allocation improves data coverage in our experiments, it does not by itself guarantee fairness, safety, or demographic representativeness. We therefore recommend auditing the candidate data, target set, and high-weight clusters before deployment.

The scores produced by ACE estimate training utility under a specified target objective. They should not be interpreted as definitive evidence of causality, data ownership, legal compliance, or the intrinsic value of an individual example. ACE reduces the computational cost of data attribution, but large-scale language-model training remains resource intensive. We report both selection overhead and training cost to make this tradeoff transparent.

REPRODUCIBILITY STATEMENT

We provide the theoretical, experimental, and implementation details required to reproduce this work. The main text and appendix define all components of ACE, including proxy-feature extraction, cluster construction, representative sampling, cluster-utility estimation, and budget allocation. They also document the datasets, preprocessing procedures, model architectures, selection budgets, training hyperparameters, evaluation protocols, hardware configurations, and efficiency accounting. Complete proofs and additional experimental details are provided in the appendix.

The accompanying anonymous repository contains the code and configurations for feature extraction, clustering, attribution scoring, data selection, pre-training, post-training, evaluation, and result aggregation. For the cluster-structure analysis, we document the candidate pool, nine clustering representations and random baseline, feature processing, projection dimensions, cluster-size settings, 15 evaluation checkpoints, and evaluation protocol across seven models. Cluster quality is measured using independently computed gradients that are not used for clustering. Post-training results are averaged over three independent runs, while pre-training settings include the corpus, token budget, optimization schedule, and hardware configuration. Missing results and settings with fewer available checkpoints are explicitly identified without interpolation. Our efficiency analysis includes proxy-feature extraction, clustering, attribution scoring, data selection, and model training, so the reported end-to-end speedups account for the additional overhead introduced by ACE.

REFERENCES

Amro Abbas, Kushal Tirumala, Dániel Simig, Surya Ganguli, and Ari S. Morcos. SemDeDup: Data-efficient learning at web-scale through semantic deduplication. *arXiv preprint arXiv:2303.09540*, 2023. doi: 10.48550/arXiv.2303.09540. URL https://arxiv.org/abs/2303.09540.

Miklós Máté Badó and Kristian Fenech. Clustered influence functions. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=Efjuj6kanc. Poster.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7432–7439, 2020.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901, 2020.

Mayee F. Chen, Nicholas Roberts, Kush Bhatia, Jue Wang, Ce Zhang, Frederic Sala, and Christopher Ré. Skill-it! a data-driven skills framework for understanding and training language models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 36000–36040. Curran Associates, Inc., 2023. doi: 10.52202/075280-1562. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/70b8505ac79e3e131756f793cd80eb8d-Abstract-Conference.html.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. PaLM: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470, 2020. doi: 10.1162/tacl_a_00317. URL https://aclanthology.org/2020.tacl-1.30/.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Junwei Deng, Yuzheng Hu, Pingbang Hu, Ting-Wei Li, Shixuan Liu, Jiachen T. Wang, Dan Ley, Qirun Dai, Benhao Huang, Jin Huang, Cathy Jiao, Hoang Anh Just, Yijun Pan, Jingyan Shen, Yiwen Tu, Weiyi Wang, Xinhe Wang, Shichang Zhang, Shiyuan Zhang, Ruoxi Jia, Himabindu Lakkaraju, Hao Peng, Weijing Tang, Chenyan Xiong, Jieyu Zhao, Hanghang Tong, Han Zhao, and Jiaqi W. Ma. A survey of data attribution: Methods, applications, and evaluation in the era of generative ai, 2025. Available at SSRN: https://ssrn.com/abstract=5451054.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, July 2024. URL https://zenodo.org/records/12608602.

Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, Evan Hubinger, Kamilė Lukošiūtė, Karina Nguyen, Nicholas Joseph, Sam McCandlish, Jared Kaplan, and Samuel R. Bowman. Studying large language model generalization with influence functions. *arXiv preprint arXiv:2308.03296*, 2023. doi: 10.48550/arXiv.2308.03296. URL https://arxiv.org/abs/2308.03296.

Yuxian Gu, Li Dong, Hongning Wang, Yaru Hao, Qingxiu Dong, Furu Wei, and Minlie Huang. Data selection via optimal control for language models. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=dhAL5fy8wS. Oral.

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

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 30016–30030, 2022.

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

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1885–1894. PMLR, 2017. URL https://proceedings.mlr.press/v70/koh17a.html.

Yongchan Kwon, Eric Wu, Kevin Wu, and James Zou. DataInf: Efficiently estimating data influence in LoRA-tuned LLMs and diffusion models. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=9m02ib92Wz.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 785–794, 2017.

Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. Deduplicating training data makes language models better. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8424–8445. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.acl-long.577. URL https://aclanthology.org/2022.acl-long.577/.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. The winograd schema challenge. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Thirteenth International Conference, KR 2012*, pp. 552–561. AAAI Press, 2012. URL https://cdn.aaai.org/ocs/4492/4492-21843-1-PB.pdf.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Gadre, Hritik Bansal, Etash Guha, Sedrick Keh, Kushal Arora, Saurabh Garg, Rui Xin, Niklas Muennighoff, Reinhard Heckel, Jean Mercat, Mayee Chen, Suchin Gururangan, Mitchell Wortsman, Alon Albalak, Yonatan Bitton, Marianna Nezhurina, Amro Abbas, Cheng-Yu Hsieh, Dhruba Ghosh, Josh Gardner, Maciej Kilian, Hanlin Zhang, Rulin Shao, Sarah Pratt, Sunny Sanyal, Gabriel Ilharco, Giannis Daras, Kalyani Marathe, Aaron Gokaslan, Jieyu Zhang, Khyathi Chandu, Thao Nguyen, Igor Vasiljevic, Sham Kakade, Shuran Song, Sujay Sanghavi, Fartash Faghri, Sewoong Oh, Luke Zettlemoyer, Kyle Lo, Alaaeldin El-Nouby, Hadi Pouransari, Alexander Toshev, Stephanie Wang, Dirk Groeneveld, Luca Soldaini, Pang Wei Koh, Jenia Jitsev, Thomas Kollar, Alexandros G. Dimakis, Yair Carmon, Achal Dave, Ludwig Schmidt, and Vaishaal Shankar. Datacomp-lm: In search of the next generation of training sets for language models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 14200–14282, 2024.

Hao Liang, Zhengyang Zhao, Meiyi Qiang, Mingrui Chen, Lu Ma, Rongyi Yu, Hengyi Feng, Shixuan Sun, Zimo Meng, Xiaochen Ma, Xuanlin Yang, Qifeng Cai, Ruichuan An, Bohan Zeng, Zhen Hao Wong, Chengyu Shen, Runming He, Zhaoyang Han, Yaowei Zheng, Fangcheng Fu, Conghui He, Bin Cui, Zhiyu Li, Weinan E, and Wentao Zhang. DataFlex: A unified framework for data-centric dynamic training of large language models. *arXiv preprint arXiv:2603.26164*, 2026. URL https://arxiv.org/abs/2603.26164.

Wanchao Liang, Tianyu Liu, Less Wright, Will Constable, Andrew Gu, Chien-Chin Huang, Iris Zhang, Wei Feng, Howard Huang, Junjie Wang, Sanket Purandare, Gokul Nadathur, and Stratos Idreos. TorchTitan: One-stop PyTorch native solution for production ready LLM pretraining. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=SFN6Wm7YBI.

Zifan Liu, Amin Karbasi, and Theodoros Rekatsinas. TSDS: Data selection for task-specific model finetuning. In *Advances in Neural Information Processing Systems*, volume 37, pp. 10117–10147, 2024. doi: 10.52202/079017-0325. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/13848b5893119ff772b69812c95914fa-Abstract-Conference.html.

Shayne Longpre, Gregory Yauney, Emily Reif, Katherine Lee, Adam Roberts, Barret Zoph, Denny Zhou, Jason Wei, Kevin Robinson, David Mimno, and Daphne Ippolito. A pretrainer's guide to training data: Measuring the effects of data age, domain coverage, quality, & toxicity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3245–3276, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.179. URL https://aclanthology.org/2024.naacl-long.179/.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4885–4901, 2020.

Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. TRAK: Attributing model behavior at scale. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 27074–27113. PMLR, 2023. URL https://proceedings.mlr.press/v202/park23c.html.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In *Advances in Neural Information Processing Systems*, volume 37, pp. 30811–30849, 2024.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. In *Advances in Neural Information Processing Systems*, volume 33, pp. 19920–19930. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/e6385d39ec9394f2f3a354d9d2b88eec-Abstract.html.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8732–8740, 2020.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 4463–4473, 2019.

Yunfan Shao, Linyang Li, Zhaoye Fei, Hang Yan, Dahua Lin, and Xipeng Qiu. Balanced data sampling for language model training with clustering. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 14012–14023. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.833. URL https://aclanthology.org/2024.findings-acl.833/.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, Toronto, Canada, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.824. URL https://aclanthology.org/2023.findings-acl.824/.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4149–4158, 2019.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Jiachen T. Wang, Tong Wu, Dawn Song, Prateek Mittal, and Ruoxi Jia. Greats: Online selection of high-quality data for llm training in every iteration. In *Advances in Neural Information Processing Systems*, volume 37, pp. 131197–131223, 2024.

Shaobo Wang, Xuan Ouyang, Tianyi Xu, Yuzheng Hu, Jialin Liu, Guo Chen, Tianyu Zhang, Junhao Zheng, Kexin Yang, Xingzhang Ren, Dayiheng Liu, and Linfeng Zhang. OPUS: Towards efficient and principled data selection in large language model pre-training in every iteration. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=lcLxGrk5N9. Spotlight.

Zige Wang, Qi Zhu, Fei Mi, Minghui Xu, Ruochun Jin, and Wenjing Yang. ClusterUCB: Efficient gradient-based data selection for targeted fine-tuning of LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 18867–18880, Suzhou, China, November 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-emnlp.1026. URL https://aclanthology.org/2025.findings-emnlp.1026/.

Alexander Wettig, Aatmik Gupta, Saumya Malik, and Danqi Chen. Qurating: Selecting high-quality data for training language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 52915–52971. PMLR, 2024.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: Selecting influential data for targeted instruction tuning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54104–54132. PMLR, 2024. URL https://proceedings.mlr.press/v235/xia24c.html.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V. Le, Tengyu Ma, and Adams Wei Yu. DoReMi: Optimizing data mixtures speeds up language model pretraining. In *Advances in Neural Information Processing Systems*, volume 36, pp. 69798–69818. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/dcba6be91359358c2355cd920da3fcbd-Abstract-Conference.html.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, 2019.

Jipeng Zhang, Yaxuan Qin, Renjie Pi, Weizhong Zhang, Rui Pan, and Tong Zhang. TAGCOS: Task-agnostic gradient clustered coreset selection for instruction tuning data. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 4686–4701, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-naacl.264. URL https://aclanthology.org/2025.findings-naacl.264/.

Wanru Zhao, Yihong Chen, Yuzhi Tang, Wentao Ma, Shengchao Hu, Shell Xu Hu, Alex Iacob, Abhinav Mehrotra, and Nicholas D. Lane. Rethinking data curation in LLM training: Online reweighting offers better generalization than offline methods. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=UFwnsmFZ6R.

A Additional Instantiations of Cluster-Level Attribution

ACE only requires a sample-level utility s_i whose cluster mean can be estimated from uniformly sampled members. For a cluster C_k of size n_k, the exact cluster utility is

$$S_k = \frac{1}{n_k} \sum_{z_i \in C_k} s_i. \tag{14}$$

Given a representative subset $R_k \subseteq C_k$ with $|R_k| = m$, we estimate the cluster score by

$$\widehat{S}_k = \frac{1}{m} \sum_{z_i \in R_k} s_i. \tag{15}$$

For gradient-interaction methods, the sample utility additionally has the form

$$s_i = \langle h_i, h_{\mathcal{T}} \rangle, \tag{16}$$

where h_i is a candidate attribution feature and $h_{\mathcal{T}}$ is the corresponding target-set feature. Linearity then allows the cluster utility to be computed from a mean attribution feature. Below, we instantiate this form for LESS and TracIn, and treat loss-based selection directly as a scalar utility. In every case, ACE leaves the underlying per-example utility unchanged; it estimates its cluster mean from representatives and performs allocation at the cluster level.

A.1 Cluster-LESS

LESS scores a candidate by its gradient alignment with a target set. Let h_i and $h_{\mathcal{T}}$ denote the candidate and target features produced by the original LESS scoring pipeline. These symbols include its prescribed projection, optimizer transformation, target aggregation, and normalization, in their original order. Keeping this preprocessing abstract is intentional: the cluster-level derivation below changes neither the LESS feature construction nor its per-example score.

The sample-level LESS score is therefore

$$s_i^{\mathrm{LESS}} = \langle h_i, h_{\mathcal{T}} \rangle. \tag{17}$$

For a cluster C_k, the exact cluster-level LESS score is defined as the average score of all cluster members:

$$\begin{aligned} S_k^{\mathrm{LESS}} &= \frac{1}{n_k} \sum_{z_i \in C_k} s_i^{\mathrm{LESS}} \\ &= \frac{1}{n_k} \sum_{z_i \in C_k} \langle h_i, h_{\mathcal{T}} \rangle. \end{aligned} \tag{18}$$

Because the inner product is linear in its first argument,

$$\begin{aligned} S_k^{\mathrm{LESS}} &= \left\langle \frac{1}{n_k} \sum_{z_i \in C_k} h_i, h_{\mathcal{T}} \right\rangle \\ &= \langle \bar{h}_k, h_{\mathcal{T}} \rangle, \end{aligned} \tag{19}$$

where

$$\bar{h}_k = \frac{1}{n_k} \sum_{z_i \in C_k} h_i \tag{20}$$

is the exact mean attribution feature of cluster C_k.

Computing $\bar{h}_k$ exactly would require extracting attribution features for all examples in the cluster. Instead, we sample m representatives uniformly without replacement and define

$$\widehat{h}_k = \frac{1}{m} \sum_{z_i \in R_k} h_i. \tag{21}$$

The resulting Cluster-LESS estimator is

$$\begin{aligned}\widehat{S}_k^{\mathrm{LESS}} &= \left\langle \widehat{h}_k, h_{\mathcal{T}} \right\rangle \\ &= \left\langle \frac{1}{m} \sum_{z_i \in R_k} h_i, h_{\mathcal{T}} \right\rangle \\ &= \frac{1}{m} \sum_{z_i \in R_k} \langle h_i, h_{\mathcal{T}} \rangle \\ &= \frac{1}{m} \sum_{z_i \in R_k} s_i^{\mathrm{LESS}}. \end{aligned} \tag{22}$$

Thus, Cluster-LESS preserves the original gradient-alignment objective. It reduces the number of objective-specific target-model evaluations from N to Km, in addition to the one-time cost of proxy-feature extraction and clustering. The end-to-end runtime in the main text includes both costs.

A.2 CLUSTER-TRACIN

TracIn estimates the influence of a candidate example by accumulating gradient interactions across a set of model checkpoints $\mathcal{C} = \{1, \ldots, T\}$.

At checkpoint t, define

$$g_i^{(t)} = \nabla_\theta \ell(z_i; \theta_t) \tag{23}$$

and

$$g_{\mathcal{T}}^{(t)} = \frac{1}{|\mathcal{T}|} \sum_{z \in \mathcal{T}} \nabla_\theta \ell(z; \theta_t). \tag{24}$$

Let $\eta_t \geq 0$ denote the learning-rate weight associated with checkpoint t. The sample-level TracIn score is

$$s_i^{\mathrm{TracIn}} = \sum_{t=1}^{T} \eta_t \left\langle g_i^{(t)}, g_{\mathcal{T}}^{(t)} \right\rangle. \tag{25}$$

For cluster C_k, the exact cluster score is

$$\begin{aligned} S_k^{\mathrm{TracIn}} &= \frac{1}{n_k} \sum_{z_i \in C_k} s_i^{\mathrm{TracIn}} \\ &= \frac{1}{n_k} \sum_{z_i \in C_k} \sum_{t=1}^{T} \eta_t \left\langle g_i^{(t)}, g_{\mathcal{T}}^{(t)} \right\rangle. \end{aligned} \tag{26}$$

Since both sums are finite, we may exchange their order:

$$\begin{aligned} S_k^{\mathrm{TracIn}} &= \sum_{t=1}^{T} \eta_t \frac{1}{n_k} \sum_{z_i \in C_k} \left\langle g_i^{(t)}, g_{\mathcal{T}}^{(t)} \right\rangle \\ &= \sum_{t=1}^{T} \eta_t \left\langle \frac{1}{n_k} \sum_{z_i \in C_k} g_i^{(t)}, g_{\mathcal{T}}^{(t)} \right\rangle. \end{aligned} \tag{27}$$

Define the exact cluster-average gradient at checkpoint t as

$$\bar{g}_k^{(t)} = \frac{1}{n_k} \sum_{z_i \in C_k} g_i^{(t)}. \tag{28}$$

Then

$$S_k^{\mathrm{TracIn}} = \sum_{t=1}^{T} \eta_t \left\langle \bar{g}_k^{(t)}, g_{\mathcal{T}}^{(t)} \right\rangle. \tag{29}$$

Using the representative subset R_k, we estimate the cluster gradient at checkpoint t by

$$\widehat{g}_k^{(t)} = \frac{1}{m} \sum_{z_i \in R_k} g_i^{(t)}. \tag{30}$$

The same representative subset R_k is used at every checkpoint, so the estimator is equivalently the mean of the concatenated features below. The resulting Cluster-TracIn estimator is

$$\widehat{S}_k^{\mathrm{TracIn}} = \sum_{t=1}^{T} \eta_t \left\langle \widehat{g}_k^{(t)}, g_{\mathcal{T}}^{(t)} \right\rangle. \tag{31}$$

To express TracIn in the unified form of Equation 16, define the concatenated candidate feature

$$h_i^{\mathrm{TracIn}} = \begin{bmatrix} \sqrt{\eta_1} g_i^{(1)} \\ \sqrt{\eta_2} g_i^{(2)} \\ \vdots \\ \sqrt{\eta_T} g_i^{(T)} \end{bmatrix} \tag{32}$$

and the concatenated target feature

$$h_{\mathcal{T}}^{\mathrm{TracIn}} = \begin{bmatrix} \sqrt{\eta_1} g_{\mathcal{T}}^{(1)} \\ \sqrt{\eta_2} g_{\mathcal{T}}^{(2)} \\ \vdots \\ \sqrt{\eta_T} g_{\mathcal{T}}^{(T)} \end{bmatrix}. \tag{33}$$

Their inner product is

$$\begin{aligned} \left\langle h_i^{\mathrm{TracIn}}, h_{\mathcal{T}}^{\mathrm{TracIn}} \right\rangle &= \sum_{t=1}^{T} \left\langle \sqrt{\eta_t} g_i^{(t)}, \sqrt{\eta_t} g_{\mathcal{T}}^{(t)} \right\rangle \\ &= \sum_{t=1}^{T} \eta_t \left\langle g_i^{(t)}, g_{\mathcal{T}}^{(t)} \right\rangle \\ &= s_i^{\mathrm{TracIn}}. \end{aligned} \tag{34}$$

Therefore, TracIn is also a gradient-interaction method of the form

$$s_i = \langle h_i, h_{\mathcal{T}} \rangle,$$

and Cluster-TracIn is obtained by averaging the concatenated features of the representative examples. The original checkpoint-weighted influence score is unchanged, while the number of candidate-side, objective-specific checkpoint–example evaluations decreases from NT to KmT.

A.3 Cluster-Loss

Loss-based selection assigns each example a scalar utility. Let s_i^{Loss} denote the score used by the sample-level baseline, including its original sign and normalization convention. No interaction-form assumption is needed: ACE directly estimates the cluster mean

$$S_k^{\mathrm{Loss}} = \frac{1}{n_k} \sum_{z_i \in C_k} s_i^{\mathrm{Loss}}, \qquad \widehat{S}_k^{\mathrm{Loss}} = \frac{1}{m} \sum_{z_i \in R_k} s_i^{\mathrm{Loss}}. \tag{35}$$

Cluster-Loss therefore retains the sample-level loss criterion while reducing objective-specific score evaluations from N to Km, in addition to the one-time proxy-feature and clustering cost. Its sampling variance is governed by the within-cluster variance of the scalar loss scores. The gradient-cosine specialization used for gradient-interaction methods is not required for this estimator.

A.4 CLUSTER-OPUS

OPUS is similar to LESS in that it scores candidates by gradient alignment. The main difference is that OPUS first transforms each candidate gradient into the update geometry induced by AdamW.

At training step t, let

$$g_i^{(t)} = \nabla_\theta \ell(z_i; \theta_t) \tag{36}$$

be the candidate gradient. Holding the AdamW state fixed while scoring the current candidate buffer, OPUS uses the data-dependent update feature

$$u_i^{(t)} = P_t^{\mathrm{AdamW}} g_i^{(t)}, \qquad P_t^{\mathrm{AdamW}} \propto \mathrm{Diag}\left(\frac{1}{\sqrt{\widehat{v}_{t-1}} + \epsilon}\right), \tag{37}$$

where $\widehat{v}_{t-1}$ is AdamW's second-moment estimate. Shared learning- rate and bias-correction factors are absorbed into the score scale.

Let Π be the projection used by OPUS and define

$$h_i^{\mathrm{OPUS},t} = \Pi\Big(u_i^{(t)}\Big), \qquad h_{\mathcal{P}}^{\mathrm{OPUS},t} = \Pi\left(\frac{1}{|\mathcal{P}_t|} \sum_{z \in \mathcal{P}_t} \nabla_\theta \ell(z; \theta_t)\right), \tag{38}$$

where $\mathcal{P}_t$ is the proxy mini-batch. Thus, OPUS applies the AdamW transformation to the candidate side, while the proxy direction remains unpreconditioned.

The sample-level optimizer-aware alignment score has the same form as LESS:

$$s_{i,t}^{\mathrm{OPUS}} = \left\langle h_i^{\mathrm{OPUS},t}, h_{\mathcal{P}}^{\mathrm{OPUS},t} \right\rangle. \tag{39}$$

For cluster C_k, linearity gives

$$\begin{aligned} S_{k,t}^{\mathrm{OPUS}} &= \frac{1}{n_k} \sum_{z_i \in C_k} s_{i,t}^{\mathrm{OPUS}} \\ &= \left\langle \frac{1}{n_k} \sum_{z_i \in C_k} h_i^{\mathrm{OPUS},t}, h_{\mathcal{P}}^{\mathrm{OPUS},t} \right\rangle \\ &= \left\langle \bar{h}_{k,t}^{\mathrm{OPUS}}, h_{\mathcal{P}}^{\mathrm{OPUS},t} \right\rangle, \end{aligned} \tag{40}$$

where $\bar{h}_{k,t}^{\mathrm{OPUS}}$ is the exact mean AdamW-transformed feature of the cluster.

Using $m_k = \min(m, n_k)$ representatives sampled uniformly without replacement, we estimate

$$\widehat{h}_{k,t}^{\mathrm{OPUS}} = \frac{1}{m_k} \sum_{z_i \in R_k} h_i^{\mathrm{OPUS},t} \tag{41}$$

and obtain

$$\begin{aligned} \widehat{S}_{k,t}^{\mathrm{OPUS}} &= \left\langle \widehat{h}_{k,t}^{\mathrm{OPUS}}, h_{\mathcal{P}}^{\mathrm{OPUS},t} \right\rangle \\ &= \frac{1}{m_k} \sum_{z_i \in R_k} s_{i,t}^{\mathrm{OPUS}}. \end{aligned} \tag{42}$$

The full OPUS rule additionally subtracts within-step redundancy. If this term is used, after selecting the history $\mathcal{A}_{t,j-1}$ we replace the proxy feature in the equations above by

$$q_{t,j} = \eta_t h_{\mathcal{P}}^{\mathrm{OPUS},t} - \eta_t^2 \sum_{z_a \in \mathcal{A}_{t,j-1}} h_a^{\mathrm{OPUS},t}. \tag{43}$$

For a fixed history, the same cluster-mean derivation applies without change.

Cluster-OPUS therefore differs from Cluster-LESS only in the attribution feature used to represent each candidate: it averages AdamW-induced update features rather than the original LESS features. Consequently, the number of candidate-side OPUS features is reduced from N_t to Km, in addition to the proxy-feature extraction and clustering costs.

B PROOF OF THEOREM 2

All expectations below are conditional on fixed cluster assignments, model parameters, target features, and sample utilities at one attribution round. Thus, the only randomness is the uniform sampling of representatives. We first give the finite-population result for an arbitrary scalar utility, then obtain the gradient-coherence bound as a specialization.

Unbiasedness. Fix a cluster C_k of size n_k and choose $1 \le m_k \le n_k$ representatives uniformly without replacement. Let

$$S_k = \frac{1}{n_k} \sum_{i \in C_k} s_i, \qquad \widehat{S}_k = \frac{1}{m_k} \sum_{i \in R_k} s_i.$$

For every $i \in C_k$, the inclusion probability is $\Pr(i \in R_k) = m_k/n_k$. Therefore,

$$\mathbb{E}[\widehat{S}_k] = \frac{1}{m_k} \sum_{i \in C_k} \frac{m_k}{n_k} s_i = S_k.$$

General finite-population variance. If $n_k = 1$, then $m_k = 1$ and the estimator is exact. We therefore assume $n_k \ge 2$ below. Define the within-cluster score variance as

$$\sigma_{s,k}^2 = \frac{1}{n_k - 1} \sum_{i \in C_k} (s_i - S_k)^2.$$

Uniform sampling without replacement gives

$$\mathrm{Var}(\widehat{S}_k) = \frac{1}{m_k} \left(1 - \frac{m_k}{n_k}\right) \sigma_{s,k}^2. \tag{44}$$

This identity applies to any scalar selection objective, including loss-based selection.

Gradient-interaction specialization. Suppose $s_i = \langle h_i, h_{\mathcal{T}} \rangle$ and define

$$\bar{h}_k = \frac{1}{n_k} \sum_{i \in C_k} h_i, \qquad D_k = \frac{1}{n_k - 1} \sum_{i \in C_k} \|h_i - \bar{h}_k\|^2.$$

Because $s_i - S_k = \langle h_i - \bar{h}_k, h_{\mathcal{T}} \rangle$, Cauchy–Schwarz yields the general dispersion bound

$$\mathrm{Var}(\widehat{S}_k) \le \frac{1}{m_k} \left(1 - \frac{m_k}{n_k}\right) \|h_{\mathcal{T}}\|^2 D_k. \tag{45}$$

When all attribution features in C_k have the same norm $\|h_i\| = H_k$ and $n_k \ge 2$, define their mean pairwise cosine similarity as

$$\rho_k = \frac{1}{n_k(n_k - 1)H_k^2} \sum_{\substack{i,j \in C_k \\ i \ne j}} \langle h_i, h_j \rangle.$$

Expanding the centered sum gives

$$\sum_{i \in C_k} \|h_i - \bar{h}_k\|^2 = (n_k - 1)H_k^2(1 - \rho_k),$$

and hence $D_k = H_k^2(1 - \rho_k)$. Substitution into Equation 45 yields

$$\mathrm{Var}(\widehat{S}_k) \le \frac{1}{m_k} \left(1 - \frac{m_k}{n_k}\right) \|h_{\mathcal{T}}\|^2 H_k^2 (1 - \rho_k). \tag{46}$$

Dataset-level aggregation. The same argument applies to any deterministic aggregation of the cluster means. Let $a_k \ge 0$, $\sum_{k=1}^K a_k = 1$, and define

$$S = \sum_{k=1}^K a_k S_k, \qquad \widehat{S} = \sum_{k=1}^K a_k \widehat{S}_k.$$

If representatives are sampled independently across clusters, then

$$\mathrm{Var}(\widehat{S}) \le \|h_{\mathcal{T}}\|^2 \sum_{k=1}^{K} \frac{a_k^2}{m_k} \left(1 - \frac{m_k}{n_k}\right) H_k^2 (1 - \rho_k). \tag{47}$$

For the balanced case in Theorem 2, where $a_k = 1/K$, $n_k = n$, $m_k = m$, and $H_k = H$, this reduces to

$$\mathrm{Var}(\widehat{S}) \le \frac{1}{Km} \left(1 - \frac{m}{n}\right) \|h_{\mathcal{T}}\|^2 H^2 (1 - \bar{\rho}), \qquad \bar{\rho} = \frac{1}{K} \sum_{k=1}^{K} \rho_k. \tag{48}$$

Equation 47 also makes the assumptions behind the simplified mean-similarity form explicit: when cluster sizes, sampling rates, or feature norms differ, their contributions are weighted rather than captured exactly by an unweighted $\bar{\rho}$.

The bounds separate two sources of reliability. Increasing the representative count reduces sampling noise, while increasing within-cluster coherence makes members more interchangeable as representatives. The finite-population factor also makes the estimator exact when every cluster member is evaluated. The empirical analysis normalizes evaluator gradients before computing cosine similarity, matching the equal-norm specialization above; for arbitrary scalar utilities, Equation 44 remains the applicable general result.

Feature normalization. In all gradient-based experiments, we ℓ_2-normalize the projected candidate and target features before scoring:

$$h_i \leftarrow \frac{h_i}{\|h_i\|_2}, \qquad h_{\mathcal{T}} \leftarrow \frac{h_{\mathcal{T}}}{\|h_{\mathcal{T}}\|_2}. \tag{49}$$

Therefore, $\|h_i\|_2 = \|h_{\mathcal{T}}\|_2 = 1$, and the equal-norm condition holds exactly with $H_k = 1$; it is enforced by preprocessing rather than assumed for the raw gradient norms. The resulting utility is cosine similarity. For unnormalized features, the general bound in Equation 45 applies instead.

Empirical structure analysis. We use at most eight representatives per cluster. Specifically, we set $m_k = \min(8, n_k)$ and sample without replacement; hence, when $n_k < 8$, all cluster members are evaluated. For every cluster with $m_k \ge 2$, we measure

$$\widehat{\rho}_k = \frac{1}{m_k(m_k - 1)} \sum_{\substack{i,j \in R_k \\ i \ne j}} \langle h_i, h_j \rangle. \tag{50}$$

Singleton clusters contain no pair and are omitted only from the pairwise coherence average. This protocol uses a common maximum evaluation budget while retaining every available member of an undersized cluster.

C Experimental Details

All paired comparisons use the same backbone, candidate pool, selected-data budget, and training pipeline. Thus, within each comparison, the data-selection policy is the only experimental factor that changes.

Post-training details. Following LESS (Xia et al., 2024), we fine-tune Llama-3.2-3B, Llama-2-7B, and Mistral-7B using LoRA (Hu et al., 2022) on the attention projection matrices $\{W_q, W_k, W_v, W_o\}$, with rank 128, $\alpha = 512$, and dropout 0.1. Training uses bf16, gradient checkpointing, and a maximum sequence length of 2,048. We train for one epoch with an effective batch size of 64 and a learning rate of 2×10^{-5}. The candidate pool contains 270,679 instruction-tuning examples from FLAN-v2, Chain-of-Thought, Dolly, and OpenAssistant-1. The target set is *LESS_eval*, consisting of 375 examples from MMLU, BBH, and TyDiQA. Dynamic selection starts after 20 warm-up steps and runs every 100 optimizer steps for four rounds. Each round selects 6,400 examples, resulting in a total budget of 25,600 examples, approximately 10% of the candidate pool.

Pre-training details. To evaluate ACE's scalability to larger data pools, we extend our experiments to pre-training. Following OPUS (Wang et al., 2026), we pre-train GPT-2 Large from scratch on FineWeb using bf16, gradient checkpointing, and a sequence length of 2,048. We train for 30B

Table 4. **OOD performance of sample- and cluster-level selection.** All methods use 25.6K examples (approximately 10% of the full training set). For each method pair, we report the evaluated sample- and cluster-level checkpoints on GPQA, GSM8K, ARC-C, and HellaSwag. Δ Avg. and Speedup are measured relative to the corresponding sample-level method. Time includes embedding, clustering, selection, and training. Within each backbone, the best score in each performance column is bold, and the row with the highest Avg. is highlighted in darker blue.

Model	Method	Variant	Data ratio	OOD performance					Efficiency		
				GPQA	GSM8K	ARC-C	HellaSwag	Avg.↑	Δ Avg.	Time↓	Speedup↑
Llama-3.2-3B	TracIn	Sample-Level	10%	29.29	21.61	47.44	74.81	43.29	–	29h20m	1.0×
	TracIn	**Cluster-Level**	10%	27.78	26.84	49.40	**75.72**	44.94	+1.65	52m	33.8×
	Loss	Sample-Level	10%	26.77	25.47	46.84	74.29	43.34	–	4h27m	1.0×
	Loss	**Cluster-Level**	10%	**31.82**	**28.35**	47.35	74.42	**45.49**	+2.14	47m	5.7×
	LESS	Sample-Level	10%	27.78	25.78	**49.49**	74.31	44.34	–	23h23m	1.0×
	LESS	**Cluster-Level**	10%	31.31	26.23	47.95	74.22	44.93	+0.59	53m	26.5×
Llama-2-7B	TracIn	Sample-Level	10%	22.22	13.19	47.78	77.09	40.07	–	36h51m	1.0×
	TracIn	**Cluster-Level**	10%	25.25	15.09	49.06	**77.60**	41.75	+1.68	1h15m	29.5×
	Loss	Sample-Level	10%	26.26	15.24	47.78	77.26	41.64	–	4h01m	1.0×
	Loss	**Cluster-Level**	10%	28.28	14.48	48.63	77.21	42.15	+0.52	1h10m	3.4×
	LESS	Sample-Level	10%	23.74	13.34	48.38	77.00	40.62	–	34h27m	1.0×
	LESS	**Cluster-Level**	10%	**28.79**	**16.00**	**49.23**	77.28	**42.83**	+2.21	1h16m	27.2×
Mistral-7B	TracIn	Sample-Level	10%	26.77	34.50	58.28	82.27	50.46	–	38h07m	1.0×
	TracIn	**Cluster-Level**	10%	32.32	41.17	**58.96**	82.14	53.65	+3.19	1h45m	21.8×
	Loss	Sample-Level	10%	29.29	**45.94**	56.91	81.90	53.51	–	2h59m	1.0×
	Loss	**Cluster-Level**	10%	31.82	43.21	57.76	81.98	53.69	+0.18	1h09m	2.6×
	LESS	Sample-Level	10%	**33.33**	39.88	57.59	**82.48**	53.32	–	26h52m	1.0×
	LESS	**Cluster-Level**	10%	32.83	45.64	58.11	81.88	**54.62**	+1.30	1h05m	24.8×

tokens, corresponding to 57,220 optimizer steps, with a global batch size of 524,288 tokens on 32 H20 GPUs. We use AdamW with learning rate 2×10^{-3}, $(\beta_1, \beta_2) = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 0.1, gradient clipping at norm 1.0, and a cosine schedule with 572 warm-up steps that decays to 10% of the peak learning rate. The target set is a class-balanced proxy containing 29,626 sequences of length 512, drawn from the downstream evaluation distribution.

Clustering and representatives. For post-training, we compute 2,048-dimensional projected gradient features with Qwen2.5-0.5B and construct the partition once using standard K-means. We use a target cluster size of 64; the exact feature extraction and clustering protocol is given in Appendix E. For pre-training, we partition FineWeb into $K = 1{,}024$ clusters using the corresponding cached proxy features. In both settings, we sample $m = 8$ representatives uniformly without replacement from each cluster and use their attribution scores to estimate its utility. Feature caches and cluster assignments are reused across selection rounds and attribution objectives whenever applicable.

Evaluation and timing. For post-training, we use LM Evaluation Harness (Gao et al., 2024) following the LESS protocol: 5-shot accuracy on MMLU (Hendrycks et al., 2021), 3-shot exact match on BBH (Suzgun et al., 2023), and 1-shot macro-averaged F1 on TyDiQA (Clark et al., 2020). We report the unweighted average across the three tasks and report means and sample standard deviations over three runs. For pre-training, we follow OPUS and evaluate zero-shot log-likelihood performance on MMLU, ANLI, HellaSwag, PIQA, SIQA, WinoGrande, ARC-Challenge, CommonsenseQA, WSC, and RACE; the reported average weights all ten tasks equally. For both post-training and pre-training, efficiency is measured by cold-start end-to-end runtime, including feature extraction, clustering where applicable, attribution scoring, data selection, and model training, while excluding final evaluation.

D Out-of-Distribution Generalization

Table 4 evaluates generalization beyond the target benchmarks used for attribution. The four disjoint OOD benchmarks cover diverse capabilities: graduate-level scientific reasoning (GPQA), multi-step mathematics (GSM8K), science question answering (ARC-C), and commonsense completion (Hel-

Table 5. Gradient-space structure at the default target cluster size $M = 64$, averaged over 15 Mistral-7B LoRA checkpoints. Sep removes cross-cluster background alignment from Intra.

Clustering representation	Intra	Inter	Sep
Qwen2.5-0.5B gradient	**0.5220**	0.0299	**0.4921**
Mistral-7B gradient	0.3365	0.0226	0.3139
BGE embedding	0.2268	0.0120	0.2148
Qwen hidden L12	0.2221	0.0143	0.2078
Qwen concat L6/12/18	0.2176	0.0133	0.2043
Qwen hidden L6	0.2121	0.0113	0.2008
Qwen hidden L18	0.1885	0.0123	0.1762
Qwen hidden L20	0.1848	0.0116	0.1731
Qwen hidden L24	0.1717	0.0106	0.1611
Random assignment	0.0205	0.0202	0.0003

laSwag). All methods select the same 25.6K examples, and the reported time includes embedding, clustering, selection, and training.

Cluster-level selection improves the OOD average in all nine method–backbone combinations, with gains of 0.18–3.19 points and speedups of 2.6–$33.8\times$. The best configurations reach averages of 45.49, 42.83, and 54.62 on Llama-3.2-3B, Llama-2-7B, and Mistral-7B, respectively. These consistent gains across distinct domains and reasoning skills suggest that ACE selects broadly useful data rather than target-specific examples.

E Details of the Cluster-Structure Analysis

This section provides the key experimental details for Section 4.2. Our goal is to determine whether low-cost clustering features recover meaningful gradient structure in the candidate data. To avoid circular evaluation, all partitions are evaluated in independently computed gradient spaces that are never used by K-means.

Data and clustering representations. We use the same $N = 270{,}679$ instruction–response examples as in the post-training experiments, drawn from FLAN-v2, Chain-of-Thought, Dolly, and OpenAssistant-1. We compare ten clustering configurations: projected gradients from Qwen2.5-0.5B and Mistral-7B; BGE base-en-v1.5 embeddings; Qwen2.5-0.5B hidden states from layers 6, 12, 18, 20, and 24; concatenated hidden states from layers 6, 12, and 18; and uniform random assignment.

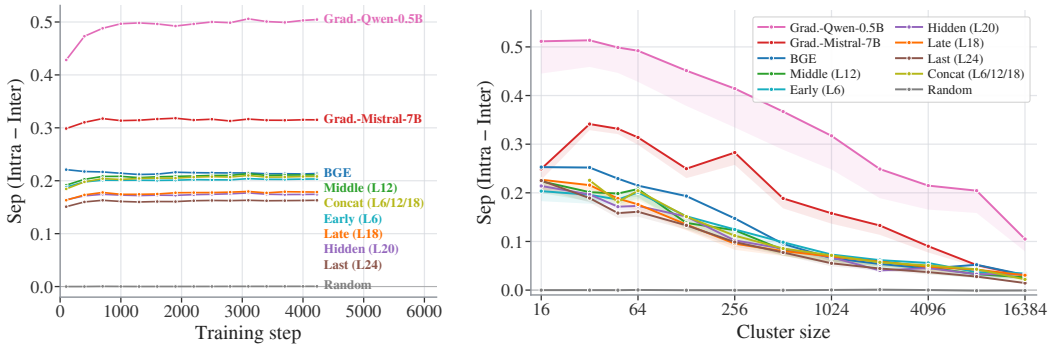
BGE embeddings are obtained from the CLS representation of the concatenated instruction and response. Qwen hidden states are mean-pooled with attention masks. For the two gradient representations, we compute per-example gradients with respect to the full q, k, v, and o projection weights, apply a $2{,}048$-dimensional TRAK Rademacher projection, and ℓ_2-normalize the result. These gradients are used only for cluster construction.

All non-random features are ℓ_2-normalized and partitioned using standard K-means with seed 3, n_init=3, and max_iter=100. For a target cluster size M, we set $K = \lceil \frac{N}{M} \rceil$. The default setting is $M = 64$, yielding $K = 4{,}230$. We additionally evaluate

$$M \in \{16, 32, 48, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384\}. \tag{51}$$

The random baseline uses the same K and assigns examples uniformly to clusters.

Effect of gradient projection. Random projection inevitably introduces approximation error and may suppress fine-grained gradient geometry. We therefore use the same $2{,}048$-dimensional projection for both gradient representations to control this factor. Although the absolute Intra and Sep values may depend on the projection dimension, Qwen2.5-0.5B gradients still substantially outperform Mistral-7B gradients under the same compression, while the larger $2{,}688$-dimensional concatenated hidden-state feature achieves only $\mathrm{Sep} = 0.2043$. Thus, projection dimension can affect measured coherence, but does not explain the observed representation ordering. We use $2{,}048$ dimensions as a practical cost–fidelity trade-off rather than claiming it to be optimal.



(a) Sep across training checkpoints. *(b)* $\overline{\text{Sep}}$ across cluster sizes.

Figure 5. Stability of background-adjusted gradient structure. **Left:** separation across Mistral-7B LoRA checkpoints at $M = 64$. **Right:** checkpoint-averaged separation across target cluster sizes; shading extends to the worst-checkpoint value and is not a confidence interval.

Independent gradient-space evaluation. We evaluate each partition using independently computed Mistral-7B LoRA gradients at 15 training checkpoints,

$$\mathcal{C}_{\text{eval}} = \{100, 400, 700, \ldots, 4000, 4230\}. \tag{52}$$

For each example, we compute gradients with respect to the LoRA parameters attached to the q, k, v, and o projections and apply a separate 8,192-dimensional TRAK projection. These evaluator gradients are never provided to K-means and are used only for structural analysis.

For each partition, we sample eight examples from all clusters containing at least eight members. The evaluated clusters and examples are fixed across checkpoints, and all reported statistics are macro-averaged over the evaluated clusters.

Within-cluster coherence and background-adjusted separation. Let $\widehat{\mathbf{g}}_i^{(t)}$ denote the unit-normalized evaluator gradient of example i at checkpoint t, let $\mathcal{S}$ be the evaluated clusters, and let R_k contain the eight sampled examples from cluster k. We measure within-cluster gradient coherence as

$$\text{Intra}_t = \frac{1}{|\mathcal{S}|} \sum_{k \in \mathcal{S}} \frac{1}{|R_k|(|R_k|-1)} \sum_{\substack{i,j \in R_k \\ i \neq j}} \left\langle \widehat{\mathbf{g}}_i^{(t)}, \widehat{\mathbf{g}}_j^{(t)} \right\rangle. \tag{53}$$

This is the empirical counterpart of the intra-cluster similarity $\bar{\rho}$ in Theorem 2.

The following analysis is auxiliary to the main-text Intra analysis. Intra is the primary quantity directly connected to the variance bound, whereas Sep tests whether the same structural conclusions remain after subtracting cross-cluster background alignment. A positive Intra value alone may partly arise from a gradient direction shared across the entire candidate corpus. We therefore compute the cross-cluster background similarity

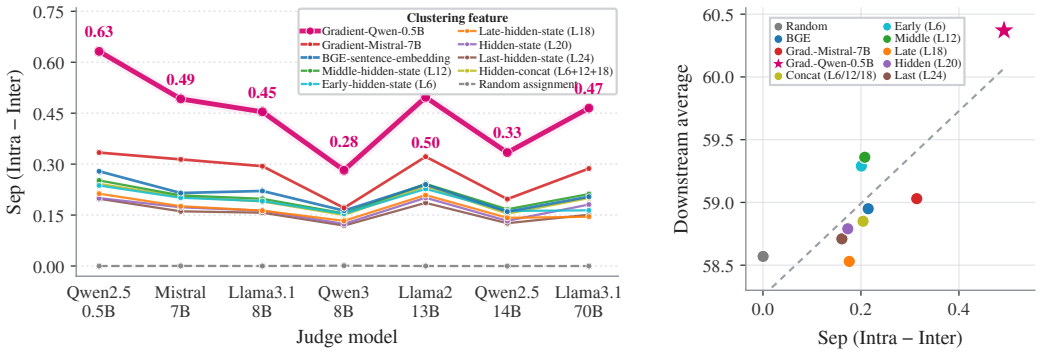
$$\text{Inter}_t = \frac{\displaystyle\sum_{\substack{k,\ell \in \mathcal{S} \\ k<\ell}} \sum_{i \in R_k} \sum_{j \in R_\ell} \left\langle \widehat{\mathbf{g}}_i^{(t)}, \widehat{\mathbf{g}}_j^{(t)} \right\rangle}{\displaystyle\sum_{\substack{k,\ell \in \mathcal{S} \\ k<\ell}} |R_k||R_\ell|}, \tag{54}$$

and define the background-adjusted separation

$$\text{Sep}_t = \text{Intra}_t - \text{Inter}_t\,. \tag{55}$$

While Intra is the theory-aligned measure of representative reliability, Sep tests whether the observed coherence is specific to the induced clusters rather than inherited from corpus-wide gradient alignment. We report checkpoint averages $\overline{\text{Intra}}$ and $\overline{\text{Sep}}$; MinSep denotes the minimum separation across checkpoints and is used only as a worst-checkpoint diagnostic.

Sep confirms intrinsic cluster-level structure. As shown in Table 5, Qwen2.5-0.5B gradients produce both the highest within-cluster coherence ($\text{Intra} = 0.5220$) and the largest background-adjusted separation ($\text{Sep} = 0.4921$). In contrast, random assignment has comparable within- and



(a) Sep across seven LoRA evaluators. *(b)* $\overline{\text{Sep}}$ vs. performance.

Figure 6. Transferability and downstream relevance of cluster structure. **Left:** clustering-feature rankings remain consistent across evaluator models. **Right:** background-adjusted separation is positively associated with run-averaged downstream performance.

cross-cluster similarities, yielding $\text{Sep} = 0.0003$. Thus, the observed coherence cannot be explained by a gradient direction shared across the corpus: it reflects genuine cluster-level structure in the candidate data.

Figure 5 further shows that this structure persists throughout training and across clustering granularities. Qwen2.5-0.5B gradient features remain strongest from the earliest evaluated checkpoint, while separation decreases only when clusters become sufficiently coarse to combine heterogeneous examples.

Transferability and downstream relevance. We evaluate the same $M = 64$ partitions with seven LoRA evaluator models ranging from Qwen2.5-0.5B to Llama-3.1-70B. Qwen2.5-0.5B gradient clustering ranks first under every evaluator, and feature rankings have Spearman correlation of at least 0.90 with the Mistral-7B reference ranking. This indicates that the identified structure is not specific to a single evaluator model.

Across the ten clustering configurations, $\overline{\text{Sep}}$ is also positively associated with downstream performance on MMLU, BBH, and TyDiQA (Pearson $r = 0.847$; Spearman $\rho = 0.794$). This association is descriptive rather than causal, but shows that stronger background-adjusted structure is consistently reflected in better data selection.

F Details of the Selection-Diversity Analysis

Selection and visualization protocol. We compare LESS and Cluster-LESS at selection step 20 using the same candidate pool and the same budget. Each method selects 1,200 unique examples and the coverage statistics below are computed from all 1,200 selected examples.

We encode each instruction–response pair with BGE-base-en-v1.5 and ℓ_2-normalize the resulting embedding. We first reduce the shared BGE embeddings to 50 dimensions with PCA and then fit a single two-dimensional t-SNE map (perplexity 50, random seed 42). Both methods are plotted using coordinates from this same fitted map; we never fit a separate projection for each method.

Coverage metrics. Let $\mathbf{z}_i \in \mathbb{R}^2$ be the shared t-SNE coordinate of example i. We divide the coordinate-wise 0.5–99.5 percentile range of the shared reference set into an 80×80 grid and let $\mathcal{G}_{\text{ref}}$ denote the cells occupied by the reference set. For a selected set S, its t-SNE coverage is

$$\text{Cov}_{\text{tSNE}}(S) = \frac{|\{g(\mathbf{z}_i) : i \in S\} \cap \mathcal{G}_{\text{ref}}|}{|\mathcal{G}_{\text{ref}}|}, \tag{56}$$

where $g(\mathbf{z}_i)$ is the grid cell containing $\mathbf{z}_i$. The shared map contains 3,012 occupied cells. LESS reaches 2,121 cells (70.4%), whereas Cluster-LESS reaches 2,323 cells (77.1%).

We additionally measure coverage before nonlinear dimensionality reduction. Let $c_{\text{BGE}}(i)$ be the cluster assigned to example i by a fixed K-means partition of the candidate pool in BGE space. We

define

$$\mathrm{Cov}_{\mathrm{BGE}}(S) = \frac{|\{c_{\mathrm{BGE}}(i) : i \in S\}|}{|\{c_{\mathrm{BGE}}(i) : i \in \mathcal{D}\}|}, \tag{57}$$

where $\mathcal{D}$ is the complete candidate pool and the denominator contains only non-empty clusters. With target cluster size 64, the retained partition contains 4,206 non-empty clusters. LESS covers 1,147 clusters (27.3%), while Cluster-LESS covers 1,563 clusters (37.2%). The quantity called *BGE-cluster diversity* in the figure is therefore an occupied-cluster coverage statistic rather than an entropy or pairwise-distance measure.

Target-domain annotation. To characterize which target-relevant regions are covered, we construct the annotation taxonomy from the target benchmarks. It consists of the 17 second-level MMLU domains together with one Multilingual category derived from the multilingual benchmark, giving 18 categories in total. We use GPT-6 as an external annotator to assign each displayed instruction–response example to one category. The annotator receives only the example text, the category definitions, and the same fixed prompt for both methods; it is not given the selection method. The right panel reports the resulting Top-1 category counts over the same 1,200 displayed examples from each method. We further manually audit the generated labels and resolve ambiguous or inconsistent cases before computing the Top-1 counts.

The shared projection and the two coverage statistics measure complementary properties: t-SNE coverage summarizes the extent reached in the shared projected space, while BGE-cluster coverage measures occupancy directly in the original embedding space. The target-domain counts then show where this additional coverage occurs relative to the evaluation domains.