GRAIN: GRADIENT-REFERENCED ALIGNMENT OF INFORMATIVE TOKENS FOR LLM SUPERVISED FINE-TUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Token-level selection has emerged as an effective approach to improving data efficiency in supervised fine-tuning (SFT) of large language models (LLMs). Its benefit hinges on the scoring signal: a useful score should reflect how much a token contributes to the target task, yet remain cheap enough to compute during training. Existing selectors score tokens by loss comparisons against reference or historical model states, which capture learning dynamics but do not explicitly encode downstream task objectives. In contrast, gradient-based influence offers a task-aware alternative, yet incurs substantial computational overhead at the token level. To fill this gap, we introduce GRAIN, an efficient task-aware token selector that approximates gradient-based influence using last-hidden-state gradients computed directly from the forward pass. Specifically, each token is scored by the gated alignment of this gradient with task directions estimated from a small held-out set, complemented by a lightweight excess-loss term computed from cached base-model losses. Together, these components enable online selection without additional backward passes or a separately trained reference model. Theoretical analysis shows that, under stated conditions, the ungated alignment term approximates first-order token-level influence. Extensive experiments demonstrate the superiority of our proposed method over existing token selectors.

1 INTRODUCTION

Supervised fine-tuning (SFT) is still the standard way to adapt large language models (LLMs) to instructions and downstream tasks (Touvron et al., 2023; Dubey et al., 2024). The procedure is straightforward: continue training a pretrained model on instruction-response pairs until its behaviour matches the target domain. Recent work shows that carefully selected instruction data can improve downstream performance without simply increasing corpus size (Chen et al., 2023; Pang et al., 2024b; Liu et al., 2023). Whole-sample selection methods score examples using perplexity, surface cues, LLM judges, or estimated effects on held-out performance (Cao et al., 2024; Zhao et al., 2024; Chen et al., 2023; Pang et al., 2024b; Xia et al., 2024). Such methods can improve the training pool (Section 2), yet treat each retained response as a single unit. Token-level selection instead allocates supervision within a response and has emerged as an effective route to greater data efficiency in SFT.

This finer granularity matters because a response typically mixes a few decisive tokens with substantial scaffolding, including formatting, routine transitions, and repeated explanations. Because these tokens need not contribute equally to downstream performance, the benefit of token-level selection hinges on its scoring signal. A useful score should reflect a token's contribution to the target task while remaining cheap enough to compute during training. Rho-1 and Token Cleaning use loss comparisons with reference models, while ssToken combines loss changes relative to historical model states with attention-based saliency (Lin et al., 2024; Pang et al., 2025; Qin et al., 2026). These loss-comparison signals capture learning dynamics, but they do not explicitly encode downstream task objectives.

In contrast, training-data influence (Koh & Liang, 2017; Pruthi et al., 2020; Xia et al., 2024) offers a task-aware ideal: it estimates how a training example would affect a held-out objective, and first-order gradient-similarity estimators such as TracIn and LESS realise this by comparing the gradient direction of a training example with that of the held-out objective, favouring examples whose updates

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.

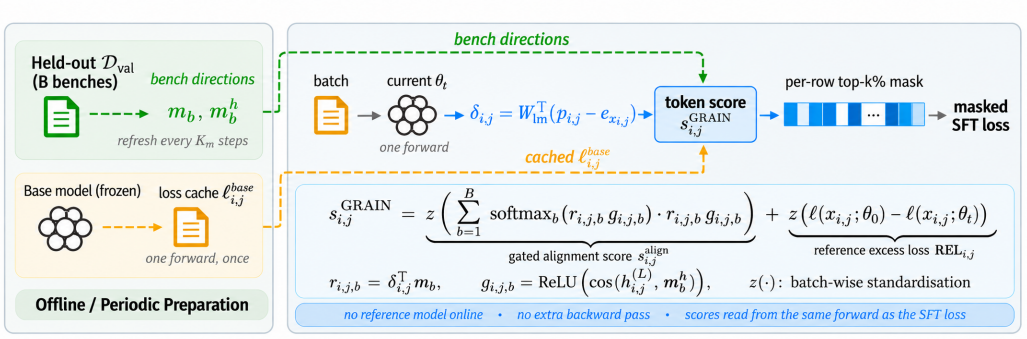


Figure 1: **Overview of GRAIN. Left:** offline and periodic preparation, namely per-bench task directions estimated on the held-out set and refreshed by periodic forward-only passes, and a one-time cache of base-model token losses. **Right:** at each SFT step, GRAIN obtains every response token's last-hidden-state gradient from the same forward pass that produces the training loss (solid path), scores it by gated alignment with the cached task directions (top, dashed) plus an excess-loss term from the cached base-model losses (bottom, dashed), and trains on a fixed proportion of top-ranked tokens within each response. No reference model is consulted online and no extra backward pass is required.

move the model toward the target task. Direct application at the token level, however, requires per-token parameter gradients and repeated held-out gradients, making it prohibitively expensive during training.

To fill this gap, we introduce GRAIN (*Gradient-Referenced Alignment of INformative tokens*), an efficient task-aware selector that approximates gradient-based influence using each token's last-hidden-state loss gradient. This gradient is available in closed form from the SFT forward pass, without extra backward computation. GRAIN scores each token by the gated alignment of this gradient with task directions estimated from a small held-out set, complemented by a lightweight excess-loss term computed from cached base-model losses. It then trains on the top-scoring tokens within each response (Figure 1). GRAIN maintains a single trainable model together with periodically refreshed task directions and a fixed base-loss cache; we refer to this as a *single-state* design. A single pretrained-model pass over the training corpus builds the base-loss cache, and periodic forward-only passes refresh the held-out task directions. Online selection therefore requires neither a separately trained reference model nor additional backward passes.

Our main contributions are as follows.

- *Efficient Task-Aware Token Selection.* We introduce GRAIN, which scores each token by gated alignment between a forward-accessible last-hidden-state gradient and task directions estimated from a small held-out set. A lightweight cached excess-loss term complements the alignment signal, enabling online selection without auxiliary-model training, online reference-model inference, or extra backward passes.
- *First-Order Influence Analysis.* Under the stated conditions, we relate the ungated alignment signal to first-order token-level influence (Proposition 1). The result provides a tractable task-aware interpretation of the alignment term.
- *Comprehensive Experiments.* Across the evaluated base models, GRAIN achieves the highest average performance among SFT methods and exceeds full-token SFT in every case, although competing methods lead on a few individual benchmarks. These results support task-aware token selection with a single trainable model.

2 RELATED WORK

Sample-level data selection. Motivated by evidence that carefully curated data can outperform much larger noisy corpora (Zhou et al., 2024), a line of work selects high-quality training subsets by completion length (Zhao et al., 2024), perplexity (Cao et al., 2024), reward or confidence scores (Gou & Nguyen, 2024; Chen & Mueller, 2024), gradient or influence signals (Zhou et al., 2023; Xia et al.,

2024), or LLM quality ratings and tags (Chen et al., 2023; Liu et al., 2023; Lu et al., 2023; Pang et al., 2024b). These methods keep or discard each response as a whole. We build on this line by training on the curated DS2 pool (Pang et al., 2024b) and asking a complementary question: which *tokens* within a curated sample carry the training signal.

Token-level selection. Rho-1 (Lin et al., 2024) originally proposed token-level filtering for pre-training, scoring each token by the loss difference between the current training model and a reference model and keeping the top-$k\%$ locally per sample. Token Cleaning (Pang et al., 2025) extended the idea to SFT, replacing per-sample local ranking with a global ranking across the dataset and introducing a self-evolving reference scheme. ssToken (Qin et al., 2026) combines loss changes relative to a historical model state with self-modulated, attention-based signals, without requiring a separately warmed-up reference. T-SHIRT (Fu et al., 2025) combines token-level informativeness (via Instruction-Following Difficulty) with a robust sample-level quality score in a two-stage selection pipeline. These methods use auxiliary model-state information or pre-computed signals. In contrast, GRAIN combines signals from the current training forward pass with cached base-model losses and periodically refreshed held-out directions, without training a separate auxiliary model.

Token-level loss reweighting. Rather than hard $0/1$ masks, an orthogonal line of work assigns continuous per-token weights. DFT (Wu et al., 2025) upweights confident tokens via stop-gradient probability scaling. TALR (Lin et al., 2025) uses exponentially decayed loss-based weights to mitigate forgetting, while EAFT (Diao et al., 2026) and RankTuner (Yu et al., 2026) calibrate weights via entropy-based criteria. Reweighting and selection are compatible: the online signals we introduce can also drive soft weighting, though we focus on the hard-mask setting for direct comparability with Rho-1 and Token Cleaning.

3 PRELIMINARIES

3.1 TOKEN-LEVEL DATA SELECTION

Consider a training set D of N prompt-response sequences and a model with parameters θ, where $x_{i,j}$ denotes the j-th token of sample i. The token-level SFT objective can be written as

$$\widehat{\mathcal{L}}_D(\theta) = \frac{1}{\sum_{(i,j)\in S} y_{i,j}} \sum_{(i,j)\in S} y_{i,j}\, \ell(x_{i,j} \mid x_{i,<j}; \theta), \tag{1}$$

where S indexes response-token positions, ℓ is the per-token cross-entropy loss, and $y_{i,j} \in \{0,1\}$ is a binary selection mask with at least one selected token. Standard SFT includes all response tokens, whereas token-level selection uses a data-dependent mask $\hat{y}_{i,j}$ to retain high-scoring tokens. The key design choice is therefore the scoring signal. Several existing methods (Lin et al., 2024; Pang et al., 2025; Qin et al., 2026) leverage per-token loss differences across distinct model states, such as a pretrained reference, a warmed-up checkpoint, or an optimisation snapshot. For example, Rho-1 defines the *excess loss* as

$$\rho(x_{i,j}; \theta, \theta') \;:=\; \ell(x_{i,j}; \theta) - \ell(x_{i,j}; \theta'), \tag{2}$$

where θ' denotes the reference model state. This discrepancy identifies tokens on which the current model performs relatively worse than the reference, but does not directly estimate how training on a token would affect a held-out objective. Moreover, depending on the method, reference-based scoring may require additional training, cached losses, or periodic state updates.

3.2 GRADIENT-SIMILARITY INFLUENCE

To connect token selection to downstream performance, we use a one-step gradient-similarity approximation, related to TracIn (Pruthi et al., 2020) and influence-guided data selection (Xia et al., 2024; Pang et al., 2024a). Suppose a small held-out set $\mathcal{D}_{\text{val}}$ is partitioned into B benchmark-specific subsets (*benches*), where $\mathcal{L}_v^{(b)}(\theta)$ denotes the average loss of bench b. At training step t, let θ_t denote the current model parameters and let $g_{i,j} := \nabla_\theta \ell(x_{i,j}; \theta_t)$ denote the token-loss gradient. The first-order influence of token $x_{i,j}$ on bench b is

$$I_b(x_{i,j} \mid \theta_t) \;:=\; -\big\langle g_{i,j},\, \nabla_\theta \mathcal{L}_v^{(b)}(\theta_t) \big\rangle, \tag{3}$$

Here $\nabla_\theta \mathcal{L}_v^{(b)}(\theta_t)$ is the bench-loss gradient. A larger $-I_b$ indicates greater predicted loss reduction. Unlike loss-based scores, influence directly captures alignment with downstream objectives without requiring a reference model. However, computing per-token gradients and repeatedly updating bench-loss gradients is prohibitively expensive. This motivates forward-accessible approximations to both gradients.

4 GRAIN: TASK-AWARE TOKEN SELECTION

GRAIN approximates gradient-similarity influence using loss gradients with respect to the final hidden state. These gradients can be computed directly from the SFT forward pass, while the corresponding held-out task directions are cached and periodically refreshed. Their alignment provides a task-aware scoring signal, further complemented by hidden-state relevance and reference excess loss.

4.1 TOKEN SCORING

Inspired by sample-level influence methods such as TracIn (Pruthi et al., 2020) and LESS (Xia et al., 2024), we explore gradient-based influence for token-level selection. However, directly computing per-token parameter gradients and held-out gradients is prohibitively expensive during training. To address this, we propose GRAIN, which approximates gradients in Eq. 3 using forward-accessible last-layer gradients, without additional backward passes. We then aggregate their gated alignment across held-out benches and incorporate reference excess loss to obtain the final token score.

Step 1: Token Gradient. Let $h_{i,j}^{(L)}$ denote the final hidden state of training token $x_{i,j}$, W_{lm} the LM head weight, and $e_{x_{i,j}}$ the one-hot vector of the target token. The gradient of the token loss with respect to its final hidden state is

$$\delta_{i,j} := \frac{\partial \ell(x_{i,j}; \theta_t)}{\partial h_{i,j}^{(L)}} = W_{\mathrm{lm}}^\top (p_{i,j} - e_{x_{i,j}}), \tag{4}$$

where $p_{i,j} = \mathrm{softmax}(W_{\mathrm{lm}} h_{i,j}^{(L)})$ is the predicted next-token distribution. Unlike the parameter gradient in Eq. 3, this last-layer gradient omits the backbone Jacobian and can be computed directly from the forward pass without an additional backward pass (Appendix B). The resulting $\delta_{i,j}$ provides an efficient surrogate for the training-token gradient. To measure its influence on downstream tasks, we next construct corresponding gradient directions from held-out examples.

Step 2: Task Directions. For each held-out bench b, we apply the same closed-form construction to its response tokens. We normalise token gradients, average them within each example, and then across examples to obtain the bench direction:

$$m_b := \frac{1}{|\mathcal{D}_{\mathrm{val}}^{(b)}|} \sum_{v \in \mathcal{D}_{\mathrm{val}}^{(b)}} \frac{1}{|S_v|} \sum_{j \in S_v} \frac{\delta_{v,j}}{\|\delta_{v,j}\|}, \tag{5}$$

where S_v denotes the response-token positions of held-out example v. We similarly construct a hidden-state direction m_b^h by averaging normalised final hidden states within and across examples, followed by normalisation of the across-example average. This aggregation gives each example equal weight while limiting the effects of response length and gradient magnitude. The resulting direction m_b serves as a cached surrogate for the held-out bench gradient in Eq. 3 and is periodically refreshed as training progresses.

Together, $\delta_{i,j}$ and m_b approximate the two sides of gradient-similarity influence. Their inner product therefore provides a last-layer measure of how well a training token aligns with each held-out task.

Step 3: Task-Aware Alignment. Given the token gradient $\delta_{i,j}$ and the cached bench direction m_b, we measure their directional agreement through an inner product. However, gradient alignment alone may produce spurious matches when a token's hidden representation is unrelated to the bench. We therefore use hidden-state similarity to gate each bench-specific alignment:

$$r_{i,j,b} := \delta_{i,j}^\top m_b, \quad g_{i,j,b} := \mathrm{ReLU}\big(\cos(h_{i,j}^{(L)}, m_b^h)\big). \tag{6}$$

Here $r_{i,j,b}$ measures gradient alignment with bench b, while $g_{i,j,b}$ suppresses matches with low representation similarity. Their product provides the gated alignment between the training token and the bench. Let z denote standardisation over the trainable tokens of the current batch. Since a token may contribute to multiple downstream tasks, we aggregate its gated alignment across all benches:

$$w_{i,j,b} := \mathrm{softmax}_b\big(r_{i,j,b}\, g_{i,j,b}\big), \qquad s_{i,j}^{\mathrm{align}} := z\Big(\sum_{b=1}^{B} w_{i,j,b}\, r_{i,j,b}\, g_{i,j,b}\Big). \tag{7}$$

The softmax is taken over the raw gated alignments with temperature one and provides a smooth weighting over benches. The standardisation uses the current-batch mean and epsilon-stabilised standard deviation; no per-bench standardisation or running statistics are used. The resulting alignment score captures directional agreement with held-out objectives, modulated by representation relevance. However, current gradient alignment alone does not account for learning progress relative to the pretrained model.

Step 4: Final Token Score. To complement task alignment, we incorporate Reference Excess Loss (REL) (Qin et al., 2026), which measures how much the current model has improved on each token relative to the pretrained base model θ_0:

$$\mathrm{REL}_{i,j} := z\left(\ell(x_{i,j}; \theta_0) - \ell(x_{i,j}; \theta_t)\right). \tag{8}$$

This signal complements alignment by capturing learned token behaviours that may otherwise receive low scores due to small current gradients. Here z is the same current-batch standardisation operator as in Eq. 7, applied separately to the base-to-current loss differences. The base-model losses are computed once offline and cached, while current losses are obtained from the SFT forward pass. Batch-wise standardisation places the alignment and REL channels on a common scale before their equal-weight sum:

$$s_{i,j}^{\mathrm{GRAIN}} := s_{i,j}^{\mathrm{align}} + \mathrm{REL}_{i,j}. \tag{9}$$

The final score jointly captures downstream task alignment and base-to-current learning progress, providing the ranking criterion for token selection during SFT.

4.2 Token Selection and Training

The final GRAIN score determines the selection mask in Eq. 1. Let k denote the selected-token proportion. To maintain this proportion across responses, we retain the top-$k\%$ tokens within each response and optimise the restricted SFT loss over each batch $\mathcal{B}$:

$$\widehat{\mathcal{L}}_{\mathcal{B}}(\theta) = \frac{1}{\sum_{i\in\mathcal{B}}\sum_{j\in S_i}\hat{y}_{i,j}} \sum_{i\in\mathcal{B}}\sum_{j\in S_i} \hat{y}_{i,j}\, \ell(x_{i,j} \mid x_{i,<j}; \theta), \tag{10}$$

where S_i denotes the response-token positions of sample i, $s_i^{\mathrm{GRAIN}} := (s_{i,j}^{\mathrm{GRAIN}})_{j\in S_i}$ is its vector of token scores, and $\hat{y}_{i,j} = \mathbb{I}[j \in \mathrm{TopK}(s_i^{\mathrm{GRAIN}}, k)]$ is the binary selection mask. Selection is performed independently for each response, so each response contributes approximately $k\%$ of its response tokens, subject to integer rounding. Unselected token labels are set to `-100` and ignored by standard cross-entropy with `ignore_index=-100`; under default mean reduction, the loss averages over all selected tokens in the batch. We use $k = 60\%$ by default, following prior work (Lin et al., 2024; Pang et al., 2025; Qin et al., 2026), and evaluate its sensitivity in Section 6 (Figure 3, left).

Training Procedure. Algorithm 1 summarises the training process. We begin with full-token SFT warmup to establish base-to-current loss differences before activating token selection. No held-out refresh occurs during warmup. Let K_m denote the held-out-direction refresh interval. We refresh the cached bench directions immediately when warmup ends and every K_m global training steps thereafter, using forward-only passes and an exponential moving average (EMA). At each selection-phase step, GRAIN computes token scores, selects the highest-scoring tokens within each response, and updates the model using the restricted objective.

Algorithm 1 GRAIN Training Loop

Require: Base model θ_0, training set $\mathcal{D}_{\text{train}}$, held-out set $\mathcal{D}_{\text{val}}$, cached base losses $\{\ell_{i,j}^{\text{base}}\}$, total steps T, warmup T_w, refresh interval $K_m = 100$, direction EMA decay $\alpha_m = 0.7$, selection ratio k

Initialise cached $\{m_b, m_b^h\}_{b=1}^{B}$ from one forward pass of θ_0 on $\mathcal{D}_{\text{val}}$
for $t = 0, \ldots, T-1$ **do**
  if $t < T_w$ **then**
    Perform a standard full-token SFT update
  else
    if $t = T_w$ or K_m global steps have elapsed since the last refresh **then**
      Forward $\mathcal{D}_{\text{val}}$ on θ_t to obtain fresh bench directions
      EMA-update m_b and m_b^h with decay α_m; renormalise m_b^h
    end if
    Sample batch $\mathcal{B}_t \subset \mathcal{D}_{\text{train}}$ and forward θ_t
    Obtain logits, token losses, and final hidden states
    Compute token gradients $\delta_{i,j}$ using Eq. 4
    Compute alignment $r_{i,j,b}$ and gate $g_{i,j,b}$ using Eq. 6
    Compute batch-standardised aggregated alignment $s_{i,j}^{\text{align}}$ using Eq. 7
    Compute batch-standardised $\text{REL}_{i,j}$ and final scores $s_{i,j}^{\text{GRAIN}}$ using Eqs. 8 and 9
    Select top-$k\%$ tokens per response and update θ_t using Eq. 10
  end if
end for
Return θ_T

4.3 THEORETICAL ANALYSIS

We theoretically investigate the connection between GRAIN's last-layer alignment and the first-order influence in Eq. 3. By the chain rule, the parameter gradient of token $x_{i,j}$ can be expressed as $g_{i,j} = J_{i,j}^{\top}\delta_{i,j}$, where $J_{i,j}$ is the Jacobian of its final hidden state with respect to the trainable parameters. This decomposition reveals that influence depends on both last-layer gradient alignment and the Jacobian interactions between training and held-out examples.

To obtain a tractable approximation, we introduce the following two assumptions on the held-out gradient directions and their Jacobian interactions.

Assumption 1 (Bench-Averaged Jacobian Approximation)**.** *The bench-averaged Jacobian kernel acts approximately as a positive scalar on the held-out gradient directions.*

Assumption 2 (Within-Bench Gradient Regularity)**.** *The magnitudes of held-out gradient contributions exhibit small variation across examples within each bench.*

Together with the *example-level collapse* approximation, which maps each held-out token-gradient sum to the per-example direction underlying Eq. 5, these assumptions allow us to approximate parameter-space influence using GRAIN's last-layer directions.

Proposition 1 (Last-Layer Influence Approximation)**.** *Under the example-level collapse approximation and Assumptions 1 and 2, the first-order influence admits the approximation*

$$I_b(x_{i,j} \mid \theta_t) \approx -c_b \kappa_{b,i,j}\, \delta_{i,j}^{\top} m_b, \tag{11}$$

where $c_b, \kappa_{b,i,j} > 0$. If $\kappa_{b,i,j}$ is approximately constant across tokens within a training step (A3), the ungated alignment approximately preserves the per-bench ranking induced by negative influence, provided the approximation errors do not alter the ordering.

Proposition 1 provides a conditional justification for GRAIN's last-layer alignment as a surrogate for gradient-similarity influence. In particular, under these assumptions, larger alignment corresponds to greater predicted reduction in the held-out bench loss. This connection explains why the inexpensive inner product $\delta_{i,j}^{\top} m_b$ can serve as a task-aware token selection signal without computing full parameter gradients. Importantly, the theoretical result applies only to the ungated per-bench alignment, rather than the complete GRAIN score. The proof, an expanded discussion of the assumptions, and their limitations are provided in Appendix B.

Table 1: Performance comparison of GRAIN and baseline methods across seven math/reasoning benchmarks (original scores $\times 100$). Among SFT methods, the best results are highlighted in **boldface** and the second-best are underlined, including ties. The Base rows are shown for reference. The default selected-token proportion is $k = 60\%$.

Method	MMLU	BBH	GSM8K	MathQA	GSM-Plus	ASDiv	Minerva	AVG
Base model: LLaMA-3.2-3B								
BASE (no SFT)	56.42	45.65	26.65	35.62	15.61	3.81	5.88	27.09
FULL TOKENS (50K)	56.72	47.96	29.33	36.06	16.67	55.00	5.48	35.32
UNIFORM RANDOM (k=60%)	56.81	47.41	27.72	36.23	17.34	55.70	3.99	35.03
RHO-1	56.74	48.15	29.94	36.64	17.55	57.27	4.19	35.78
TOKEN CLEANING	53.14	43.89	28.10	**37.96**	14.35	55.17	5.98	34.08
SSTOKEN	56.81	43.06	27.11	36.50	14.35	49.08	6.08	33.28
GRAIN (ours)	**56.86**	**48.30**	**31.32**	37.31	**19.65**	**58.02**	**7.68**	**37.02**
Base model: LLaMA-3.1-8B								
BASE (no SFT)	65.23	61.94	49.62	42.46	31.69	4.51	17.45	38.99
FULL TOKENS (50K)	65.78	63.06	54.52	42.66	35.44	68.67	19.84	50.00
UNIFORM RANDOM (k=60%)	64.88	62.45	52.18	42.33	33.68	63.17	18.74	48.20
RHO-1	65.18	63.89	57.58	42.83	**36.32**	71.08	19.64	50.93
TOKEN CLEANING	65.41	64.17	57.66	42.12	36.21	71.34	18.44	50.76
SSTOKEN	65.23	61.48	55.28	**43.00**	35.02	**71.74**	16.85	49.80
GRAIN (ours)	**65.99**	**64.72**	**57.97**	42.92	**36.32**	71.53	**19.94**	**51.34**
Base model: Qwen2.5-7B								
BASE (no SFT)	74.21	64.07	78.86	51.96	52.80	12.80	34.39	52.73
FULL TOKENS (50K)	74.25	63.91	78.94	52.78	56.19	**82.21**	33.50	63.11
UNIFORM RANDOM (k=60%)	73.51	63.79	77.32	51.98	55.49	81.25	34.48	62.55
RHO-1	73.69	63.70	75.27	51.83	52.65	81.55	29.91	61.23
TOKEN CLEANING	73.80	64.07	74.81	52.13	52.78	81.20	28.02	60.97
SSTOKEN	74.16	60.65	74.35	53.15	52.36	81.46	33.50	61.38
GRAIN (ours)	**74.42**	**64.17**	**79.48**	**53.31**	**57.41**	81.73	**35.09**	**63.66**

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Data Pool We use a high-quality data pool of 50,000 samples drawn from five popular SFT datasets: Flan-v2 (Longpre et al., 2023), Open-Assistant 1 (Köpf et al., 2023), Stanford Alpaca (Taori et al., 2023), Dolly (Databricks, 2023), and WizardLM (Xu et al., 2023). The pool is constructed by the DS^2 quality-rating pipeline (Pang et al., 2024b), which selects high-quality samples using quality-rating scores generated by LLMs. The resulting pool is also used by Token Cleaning (Pang et al., 2025), providing a common data basis for comparisons. Dataset composition and token-length statistics are provided in Appendix D.1. For the reference-based baselines, we use the first 10,000 samples as a warmup subset.

Base Models In this paper, we select three popular open-source LLMs as our base models: `LLaMA-3.2-3B`, `LLaMA-3.1-8B` (Dubey et al., 2024), and `Qwen2.5-7B` (Yang et al., 2024). We fine-tune each model with low-rank adaptation (LoRA) on samples from the data pool. Specifically, we warm up by fine-tuning the base model on the 10k-sample subset with full tokens to obtain the initial reference for RHO-1 and TOKEN CLEANING. SSTOKEN and GRAIN need no reference warmup; GRAIN's own full-token selection warmup covers the first 15% of training steps. Full hyperparameters are provided in Appendix D.3.

Baselines There are several baselines for performance comparisons: 1) BASE denotes the pretrained base model without SFT; 2) FULL TOKENS fine-tunes it using all response tokens from the data pool; 3) UNIFORM RANDOM randomly selects $k\%$ of response tokens per example without replacement; 4) RHO-1 (Lin et al., 2024) ranks tokens by excess loss against a warmup reference and selects the top-$k\%$; 5) SSTOKEN (Qin et al., 2026) uses attention-based saliency with self-modulated weighting and the pretrained base model as its historical state; 6) TOKEN CLEANING (Pang et al., 2025) uses a self-evolving reference, refreshing it and recomputing per-token losses over four rounds.

Held-out Set GRAIN uses a held-out set of 583 examples drawn from six benches, roughly 100 per bench and all disjoint from the corresponding evaluation test items; this is about 1% of the combined source splits (roughly 51k examples). No held-out refresh occurs during the selection warmup. We refresh the cached task directions immediately when warmup ends and every 100 global training steps thereafter, using an EMA decay of 0.7. More details are provided in Appendix C.

Evaluation To comprehensively evaluate token selection methods, we adopt seven popular benchmarks, including MMLU (Hendrycks et al., 2020), BBH (Suzgun et al., 2022), GSM8K (Cobbe et al., 2021), MATHQA (Amini et al., 2019), GSM-PLUS-MINI (Li et al., 2024), ASDIV (Miao et al., 2020), and MINERVA-MATH (Lewkowycz et al., 2022). These benchmarks assess knowledge, broad reasoning, and mathematical problem solving across multiple-choice and generative tasks. Task performance is evaluated using `lm-evaluation-harness`[1].

5.2 MAIN RESULTS

GRAIN achieves the highest AVG across all three base models. Table 1 shows that GRAIN achieves the highest average performance across all three base models, outperforming full-token SFT, uniform random selection, and existing token selection methods. It also achieves the highest scores on most individual benchmarks, although competing methods lead on a few benchmarks.

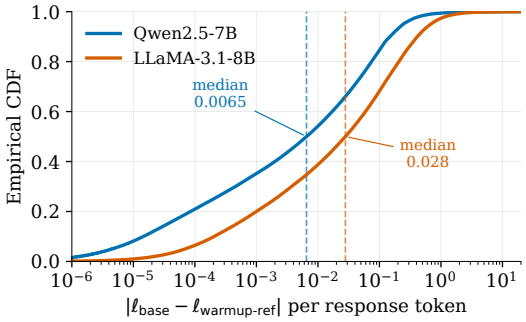


Figure 2: **Reference signals are compressed on Qwen.** Empirical CDF of the per-token reference signal on DS2-50k (3M tokens per model). The Qwen2.5-7B distribution is shifted toward zero relative to LLaMA-3.1-8B (median $4.3\times$ smaller).

Compressed Reference Signals on Qwen2.5-7B. As shown in Table 1, reference-based selectors underperform full-token SFT on Qwen2.5-7B, whereas GRAIN achieves higher performance. Figure 2 shows that Qwen2.5-7B has smaller base-to-reference loss differences than LLaMA-3.1-8B, making loss-based token rankings more sensitive to perturbations. While this does not establish a causal explanation, the results suggest that incorporating held-out gradient alignment can complement loss-based signals.

6 ABLATION STUDY

We investigate the contributions of GRAIN's scoring components, its sensitivity to the selected-token proportion and to the held-out set size, and its computational cost. Unless otherwise specified, experiments use LLaMA-3.2-3B on the 7-benchmark math/reasoning suite.

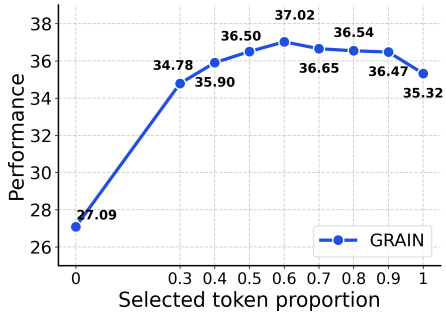
Contribution of individual scoring signals. To understand the contribution of each scoring channel, we compare the alignment and REL components individually (ALIGN-ONLY and REL-ONLY) against the full GRAIN score. All variants use the same training configuration and selection ratio of $k = 60\%$. Table 2 shows that combining alignment and REL achieves the highest performance across all benchmarks. While alignment alone performs better on most benchmarks, REL provides complementary signals, particularly on ASDiv. Their combination consistently outperforms either component alone, highlighting the benefit of integrating both scoring signals.

GRAIN performs best at the default selection ratio. Figure 3 (left) shows the effect of the selected-token proportion on downstream performance, with the scorer and held-out set fixed. Performance rises sharply above the pretrained base model, continues improving up to the default selection ratio, and then declines gradually as more tokens are retained. Most tested selective settings outperform full-token SFT, but the lowest tested retention rate does not. The broad plateau around the optimum indicates moderate sensitivity rather than invariance to the selection ratio.

[1] `https://github.com/EleutherAI/lm-evaluation-harness`

Table 2: Ablation of GRAIN's scoring components on the 7-benchmark math/reasoning suite (scores $\times 100$). ALIGN-ONLY uses gated gradient alignment, REL-ONLY uses reference excess loss, and GRAIN combines both channels. The token selection ratio is $k = 60\%$. The best result in each column is highlighted in bold.

Variant	MMLU	BBH	GSM8K	MathQA	GSM-Plus	ASDiv	Minerva	AVG
REL-ONLY	55.30	43.43	27.72	37.28	15.53	55.08	5.68	34.29
ALIGN-ONLY	56.66	48.06	30.93	36.67	17.80	10.25	5.68	29.44
GRAIN	**56.86**	**48.30**	**31.32**	**37.31**	**19.65**	**58.02**	**7.68**	**37.02**



Method	Online: Step (s)	Online: Train (min)	Online: Overhead (%)	Offline: Ref. (min)	Offline: Cache (min)	Total (min)
FULL TOKENS	4.80	83	—	—	—	83
RHO-1	4.85	84	1	15	5	104
SSTOKEN	4.99	87	4	—	5	92
TOKEN CLEANING	4.85	336	1	15	25	376
GRAIN (ours)	5.86	102	22	—	5	107

Figure 3: **Left:** Sensitivity of GRAIN to the selected-token proportion on LLaMA-3.2-3B; $k = 0$ denotes the pretrained base model without SFT. **Right:** Wall-clock time comparison of GRAIN and baseline methods on DS2-50k using $4\times$A5000 GPUs.

GRAIN maintains competitive end-to-end efficiency. Figure 3 (right) compares the online training overhead, offline preparation, and end-to-end wall-clock cost of GRAIN and the baselines under the training configuration of Section 5.1. GRAIN incurs higher online overhead than baselines that reuse cached scores, since the closed-form last-layer-gradient computation runs at every step, but its reference channel uses the pretrained base model and avoids reference-model warmup and repeated rescoring. As a result, its total cost stays close to Rho-1 and well below the four-round Token Cleaning reproduction, while Full Tokens and ssToken remain cheaper overall.

GRAIN benefits modestly from a larger held-out set. Table 3 varies the number of held-out examples per bench, reported together with the total sample size over the six held-out benches (ASDiv has no held-out split), while keeping the scorer and all training settings fixed. Performance improves monotonically with held-out-set size, and the default configuration performs best. The smaller configurations remain close to the default, indicating that useful benchmark directions can be estimated from fewer examples, but the results do not support complete insensitivity to held-out-set size.

Table 3: Sensitivity of GRAIN to the held-out size on LLaMA-3.2-3B.

Per bench	Total Size	AVG
$\sim$25	150	36.26
$\sim$50	300	36.95
$\sim$100 (default)	583	37.02

7 CONCLUSION

In this work, we address the challenge of identifying informative tokens for data-efficient LLM supervised fine-tuning. While token-level gradient influence offers a principled measure of downstream task relevance, computing full parameter gradients for individual tokens is prohibitively expensive. Existing loss-based approaches provide cheaper alternatives but do not explicitly capture how each token contributes to downstream objectives. To bridge this gap, we propose GRAIN, which estimates task-aware token relevance by aligning last-hidden-state gradients with benchmark-specific directions derived from a small held-out set. We further incorporate reference excess loss to complement task alignment with information about training progress. This design enables online token selection without additional backward passes or a separate reference model. Experiments across multiple base models demonstrate consistent performance improvements over baselines.

AI USE STATEMENT

In this work, we used generative AI tools to assist with code implementation, experiment tooling, and editing of the manuscript text. All technical ideas, experimental designs, and claims were conceived and verified by the authors, who take full responsibility for the content of this paper.

ETHICS STATEMENT

This work studies token-level data selection for supervised fine-tuning on publicly available datasets and benchmarks; it does not involve human subjects, private data, or applications with foreseeable direct negative societal impact.

REPRODUCIBILITY STATEMENT

All datasets and benchmarks used are public. Section 5.1 and Appendix D specify the training and evaluation configurations. Appendix C describes the held-out set construction. Appendix E details the closed-form gradient computation, score standardisation, and token-selection implementation. We will release the code and held-out set upon publication.

REFERENCES

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2357–2367. Association for Computational Linguistics, 2019. doi: 10.18653/v1/N19-1245.

Yihan Cao, Yanbin Kang, Chi Wang, and Lichao Sun. Instruction mining: Instruction data selection for tuning large language models. In *First Conference on Language Modeling*, 2024.

Jiuhai Chen and Jonas Mueller. Automated data curation for robust language model fine-tuning. *arXiv preprint arXiv:2403.12776*, 2024.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, et al. AlpaGasus: Training a better alpaca with fewer data. *arXiv preprint arXiv:2307.08701*, 2023.

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

Databricks. Free Dolly: Introducing the world's first truly open instruction-tuned LLM, 2023.

Muxi Diao, Lele Yang, Wuxuan Gong, Yutong Zhang, Zhonghao Yan, Yufei Han, Kongming Liang, Weiran Xu, and Zhanyu Ma. Entropy-adaptive fine-tuning: Resolving confident conflicts to mitigate forgetting. *arXiv preprint arXiv:2601.02151*, 2026.

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

Yanjun Fu, Faisal Hamman, and Sanghamitra Dutta. T-SHIRT: Token-selective hierarchical data selection for instruction tuning. *arXiv preprint arXiv:2506.01317*, 2025.

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. A framework for few-shot language model evaluation, 12 2023. URL `https://zenodo.org/records/10256836`.

Qi Gou and Cam-Tu Nguyen. Mixed preference optimization: Reinforcement learning with data selection and better reference model. *arXiv preprint arXiv:2403.19443*, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70, pp. 1885–1894. PMLR, 2017.

Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Duc Nguyen, Oliver Stanley, Richárd Nagyfi, et al. Openassistant conversations-democratizing large language model alignment. *Advances in Neural Information Processing Systems*, 36, 2023.

Jaehoon Lee, Lechao Xiao, Samuel S. Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 3843–3857, 2022.

Qintong Li, Leyang Cui, Xueliang Zhao, Lingpeng Kong, and Wei Bi. GSM-Plus: A comprehensive benchmark for evaluating the robustness of LLMs as mathematical problem solvers. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2961–2984. Association for Computational Linguistics, 2024.

Jiacheng Lin, Zhongruo Wang, Kun Qian, Tian Wang, Arvind Srinivasan, Hansi Zeng, Ruochen Jiao, Xie Zhou, Jiri Gesi, Dakuo Wang, Yufan Guo, Kai Zhong, Weiqi Zhang, Sujay Sanghavi, Changyou Chen, Hyokun Yun, and Lihong Li. SFT doesn't always hurt general capabilities: Revisiting domain-specific fine-tuning in LLMs. *arXiv preprint arXiv:2509.20758*, 2025. ICLR 2026.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Yelong Shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, and Weizhu Chen. Not all tokens are what you need for pretraining. *arXiv preprint arXiv:2404.07965*, 2024. NeurIPS 2024.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. *arXiv preprint arXiv:2312.15685*, 2023.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, and Adam Roberts. The Flan collection: Designing data and methods for effective instruction tuning. In *ICML*, 2023.

Keming Lu, Hongyi Yuan, Zheng Yuan, Runji Lin, Junyang Lin, Chuanqi Tan, Chang Zhou, and Jingren Zhou. # instag: Instruction tagging for analyzing supervised fine-tuning of large language models. In *The Twelfth International Conference on Learning Representations*, 2023.

Sadhika Malladi, Alexander Wettig, Dingli Yu, Danqi Chen, and Sanjeev Arora. A kernel-based view of language model fine-tuning. In *International Conference on Machine Learning (ICML)*, 2023.

Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. A diverse corpus for evaluating and developing english math word problem solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 975–984, 2020.

Jinlong Pang, Jialu Wang, Zhaowei Zhu, Yuanshun Yao, Chen Qian, and Yang Liu. Fairness without harm: An influence-guided active sampling approach. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024a.

Jinlong Pang, Jiaheng Wei, Ankit Parag Shah, Zhaowei Zhu, Yaxuan Wang, Chen Qian, Yang Liu, Yujia Bao, and Wei Wei. Improving data efficiency via curating LLM-driven rating systems. *arXiv preprint arXiv:2410.10877*, 2024b.

Jinlong Pang, Na Di, Zhaowei Zhu, Jiaheng Wei, Hao Cheng, Chen Qian, and Yang Liu. Token cleaning: Fine-grained data selection for LLM supervised fine-tuning. *arXiv preprint arXiv:2502.01968*, 2025. ICML 2025.

Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. TRAK: Attributing model behavior at scale. In *International Conference on Machine Learning (ICML)*, 2023.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 19920–19930, 2020.

Xiaohan Qin, Xiaoxing Wang, Ning Liao, Cancheng Zhang, Xiangdong Zhang, Mingquan Feng, Jingzhi Wang, and Junchi Yan. ssToken: Self-modulated and semantic-aware token selection for LLM fine-tuning. In *International Conference on Learning Representations (ICLR)*, 2026. https://openreview.net/forum?id=wZ4UoRKDNa.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, et al. Challenging BIG-Bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

Rohun Taori, Ishaan Gulrajani, Ting Zhang, Yann Dubois, Xiaodan Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023. GitHub repository.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. LLaMA-2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Chandu, David Wadden, Kelsey MacMillan, Noah A Smith, Iz Beltagy, and Hannaneh Hajishirzi. How far can camels go? exploring the state of instruction tuning on open resources. In *Advances in Neural Information Processing Systems*, 2023.

Yongliang Wu, Yizhou Zhou, Ziheng Zhou, Yingzhe Peng, Xinyu Ye, Xinting Hu, Wenbo Zhu, Lu Qi, Ming-Hsuan Yang, and Xu Yang. On the generalization of SFT: A reinforcement learning perspective with reward rectification. *arXiv preprint arXiv:2508.05629*, 2025. ICLR 2026.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: Selecting influential data for targeted instruction tuning. *arXiv preprint arXiv:2402.04333*, 2024. ICML 2024.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. WizardLM: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*, 2023.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Wenhao Yu, Shaohang Wei, Jiahong Liu, Yifan Li, Minda Hu, Aiwei Liu, Hao Zhang, and Irwin King. Probability-entropy calibration: An elastic indicator for adaptive fine-tuning. *arXiv preprint arXiv:2602.01745*, 2026.

Hao Zhao, Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. Long is more for alignment: A simple but tough-to-beat baseline for instruction fine-tuning. *arXiv preprint arXiv:2402.04833*, 2024.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. LIMA: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36, 2024.

Daquan Zhou, Kai Wang, Jianyang Gu, Xiangyu Peng, Dongze Lian, Yifan Zhang, Yang You, and Jiashi Feng. Dataset quantization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17205–17216, 2023.

Appendix

Organisation of the Appendix

The Appendix is organised as follows.

- Section A discusses the scope limitations of GRAIN and the regimes in which its direction-matching surrogate may be unreliable.
- Section B proves Proposition 1 and explains the assumptions behind the last-layer influence approximation.
- Section C describes the design and composition of the held-out set used to construct benchmark directions.
- Section D specifies the training data, evaluation suite, and hyperparameters.
- Section E documents the scoring and implementation details.

A WHEN GRAIN WORKS AND WHEN IT DOES NOT

This section delineates the scope of GRAIN and identifies regimes in which its benchmark directions may be unreliable. The direction-matching surrogate in Proposition 1 is most plausible when examples within a target benchmark share a coherent gradient direction, as often occurs in math and structured reasoning. Broad factual and knowledge benchmarks such as TruthfulQA, BoolQ, and TriviaQA may instead induce heterogeneous fact-retrieval gradients. This heterogeneity defines a current scope limitation of the direction-matching surrogate.

B NTK-STYLE JUSTIFICATION OF THE LAST-LAYER SURROGATE

Overview. This section proves Proposition 1 and expands the conditions and limitations of Assumptions 1 and 2. The proof retains the per-example direction used by Eq. 5 and makes the example-level collapse approximation explicit. All conclusions remain conditional on these approximations and their residual errors.

The two main-text assumptions, the bench-averaged Jacobian approximation (A1; Assumption 1) and within-bench gradient regularity (A2; Assumption 2), are *bench-averaged empirical ansatzes* on the local linearisation of the network around θ_t. They are not direct consequences of NTK theory, which establishes the kernel only in the infinite-width / strict-lazy limit (Jacot et al., 2018; Lee et al., 2019); rather, they are spiritually related to that line of work and to local-linearisation arguments already in use for influence-style data selection at LLM scale: TRAK (Park et al., 2023) (linearised model around θ_t for attribution), LESS (Xia et al., 2024) (gradient-similarity influence over LoRA), and Pang et al. (2024a) (gradient-similarity sampling). Malladi et al. (2023) provides the closest formal support, arguing a kernel/lazy regime for prompt-tuning-style fine adaptation; we do not claim it directly transfers to LoRA at our scale.

Proof of Proposition 1. For each held-out example v, define its per-example direction by

$$u_v \;:=\; \frac{1}{|S_v|}\sum_{a\in S_v}\frac{\delta_{v,a}}{\|\delta_{v,a}\|}, \qquad m_b \;=\; \mathbb{E}_{v\in\mathcal{D}_{\text{val}}^{(b)}}[u_v].$$

This is exactly the aggregation in Eq. 5; in particular, u_v is an average of unit token gradients and need not itself have unit norm. The training-token gradient satisfies

$$g_{i,j} \;=\; \nabla_\theta \ell(x_{i,j};\theta_t) \;=\; J_{i,j}^\top \delta_{i,j}, \qquad J_{i,j} \;:=\; \frac{\partial h_{i,j}^{(L)}}{\partial \theta}.$$

Example-level collapse approximation. The exact parameter gradient of the mean response loss for held-out example v is

$$g_v \;:=\; \frac{1}{|S_v|}\sum_{a\in S_v} J_{v,a}^\top \delta_{v,a}.$$

We approximate this token-level Jacobian-gradient sum by $g_v \approx r_v J_v^\top u_v$, where $r_v \geq 0$ is an effective example magnitude and J_v is an effective example-level Jacobian. Equivalently, define the collapse residual by

$$g_v \;=\; r_v J_v^\top u_v + e_v^{\mathrm{col}}.$$

Averaging over bench b and using Eq. 3 gives

$$\begin{aligned} I_b(x_{i,j} \mid \theta_t) &= -\delta_{i,j}^\top J_{i,j} \mathbb{E}_v[g_v] \\ &= -\delta_{i,j}^\top \mathbb{E}_v\left[r_v J_{i,j} J_v^\top u_v\right] + \varepsilon_{b,i,j}^{\mathrm{col}}, \end{aligned} \tag{12}$$

where $\varepsilon_{b,i,j}^{\mathrm{col}} := -\delta_{i,j}^\top J_{i,j} \mathbb{E}_v[e_v^{\mathrm{col}}]$. Thus every last-layer kernel has the dimensionally consistent form $J_{i,j} J_v^\top$.

(A1) Bench-averaged operator scalarisation (Assumption 1). Let $A_v := J_{i,j} J_v^\top$. The strict NTK limit makes this matrix-valued kernel deterministic and input-pair-dependent, but does not imply $J_{i,j} \approx J_v$ for different inputs. Linearisation only states that each input-specific Jacobian changes little as the parameters evolve. Assumption 1 is therefore an operator-form ansatz on the bench average: for some $\kappa_{b,i,j} > 0$,

$$q_{b,i,j} \;:=\; \mathbb{E}_v\left[\big(A_v - \kappa_{b,i,j} I\big) u_v\right] \;\approx\; 0, \tag{13}$$

and hence

$$\mathbb{E}_v[A_v u_v] \;=\; \kappa_{b,i,j} m_b + q_{b,i,j}.$$

This is plausible when the held-out set is task-homogeneous within a bench: individual J_v may differ, while their bench-conditional action on the directions u_v concentrates. Our $\mathcal{D}_{\mathrm{val}}$ construction (Appendix C) is designed to promote this regime.

(A2) Small within-bench magnitude spread (Assumption 2). Let $c_b := \mathbb{E}_v[r_v]$. The magnitude-weighted expectation decomposes as

$$\begin{aligned} \mathbb{E}_v[r_v A_v u_v] &= c_b\, \mathbb{E}_v[A_v u_v] + \varepsilon_{b,i,j}^{\mathrm{norm}}, \\ \varepsilon_{b,i,j}^{\mathrm{norm}} &:= \mathbb{E}_v[(r_v - c_b) A_v u_v], \\ \left\|\varepsilon_{b,i,j}^{\mathrm{norm}}\right\| &\leq M_{b,i,j}\, \mathbb{E}_v|r_v - c_b|, \qquad M_{b,i,j} := \sup_v \|A_v\|_{\mathrm{op}}, \end{aligned} \tag{14}$$

where the bound uses $\|u_v\| \leq 1$. Assumption 2 makes this residual small; it does not replace u_v by a normalised aggregate example gradient. The per-bench in-distribution items, long-CoT completions, and ICL-matched format are intended to limit this magnitude spread.

Combining the collapse approximation with Eqs. 13 and 14 yields

$$\begin{aligned} I_b(x_{i,j} \mid \theta_t) &= -c_b \kappa_{b,i,j}\, \delta_{i,j}^\top m_b + \varepsilon_{b,i,j}, \\ \varepsilon_{b,i,j} &:= \varepsilon_{b,i,j}^{\mathrm{col}} - \delta_{i,j}^\top \left(\varepsilon_{b,i,j}^{\mathrm{norm}} + c_b q_{b,i,j}\right). \end{aligned} \tag{15}$$

This gives the approximation in Proposition 1.

Within-step ranking, not cross-step magnitudes. The factor c_b is common to the tokens scored against bench b within one step. Under the proposition's additional within-step condition (A3), $\kappa_{b,i,j}$ also varies little across those tokens. Consequently, $\delta_{i,j}^\top m_b$ approximately preserves the per-bench ranking induced by $-I_b$, up to the aggregate error $\varepsilon_{b,i,j}$ and the residual variation in $\kappa_{b,i,j}$. The argument does not preserve cross-step magnitudes, but GRAIN applies its rank-based top-$k\%$ selection within each step (Eq. 10).

Failure modes and the limit of the surrogate. The argument identifies four ways the approximation can fail. (F0) *large example-level collapse error*: token-specific Jacobians and gradient magnitudes cannot be represented by a single $r_v J_v^\top u_v$, making $\varepsilon_{b,i,j}^{\mathrm{col}}$ large. (F1) *strong J_v heterogeneity within a bench*: broad factual recall items may activate qualitatively different parameter sub-circuits, so Eq. 13 admits no accurate common scalar $\kappa_{b,i,j}$; this matches the empirical scope analysis in Appendix A. (F2) *large within-bench magnitude variation*: mixed response lengths or formats can produce widely varying r_v, increasing the error in Eq. 14; the unweighted mean m_b then poorly represents the magnitude-weighted expectation. We mitigate this failure with a uniform long-CoT, ICL-matched held-out construction (Appendix C). (F3) *large within-step variation of $\kappa_{b,i,j}$*: such variation can change the token ranking even when the other residuals are small. This condition is the most closely tied to the NTK-style approximation and the hardest to verify directly.

Table 4: Per-benchmark composition of the held-out set $\mathcal{D}_{\text{val}}$, whose 583 examples come from sources disjoint from the corresponding evaluation test items. ASDiv is evaluated but excluded because it has no disjoint split.

Benchmark	Source		Held-out set	
	Split	Size	Size	Selection
MMLU	dev	285	112	2 per subject; 1-shot leave-one-out
BBH	cot-prompts	81	81	3 long-CoT exemplars per task
MathQA	train	29,837	90	rationale-consistency filter
GSM8K	train	7,473	100	long-CoT solutions
GSM-Plus	test minus testmini	8,152	100	question-hash exclusion
MATH	train	5,549	100	5 categories; solution field
Total		∼51,400	583	

C Held-Out Set Design

This section describes the construction of $\mathcal{D}_{\text{val}}$, including its three design criteria, the composition of the default $\mathcal{D}_{\text{val}}^{\text{clean}}$, and the MathQA rationale filter. Table 3 in Section 6 separately examines sensitivity to the size of $\mathcal{D}_{\text{val}}$.

C.1 Design Criteria

(D1) Disjoint from evaluation test items. If held-out items overlap with evaluation items, $\delta_{i,j} \cdot m_b$ leaks information about the test distribution into the selector. We use train/dev splits, prompt-exemplar files (BBH `cot-prompts`), or precise stable-id exclusion (GSM-Plus `test \ testmini`); benches with no clean disjoint split (ASDiv) are dropped from $\mathcal{D}_{\text{val}}$ and treated as transfer eval only.

(D2) Long chain-of-thought completions. For benchmarks evaluated through multi-step generation, $\mathcal{D}_{\text{val}}$ uses chain-of-thought (CoT) completions rather than short answer extracts. Because the example direction averages completion-token gradients, a single answer letter or number would bias it toward answer-position features rather than reasoning content.

(D3) In-context learning (ICL) prompts matching inference. Held-out prompts use leave-one-out few-shot exemplars from the same benchmark to approximate the inference format used by `lm-evaluation-harness`. The shot count matches the evaluation default when the source split permits it; exceptions are documented below.

Legacy and clean designs. Short answer-only completions and evaluation-set overlap in the legacy 16-benchmark $\mathcal{D}_{\text{val}}^{\text{legacy}}$ motivated criteria (D1) to (D3) and the clean rebuild. Table 4 lists the composition of the clean set, and Table 3 reports a size ablation for the clean $\mathcal{D}_{\text{val}}$. A controlled comparison of the two designs under the final GRAIN scorer remains future work.

C.2 Per-Benchmark Composition

ASDiv is dropped from $\mathcal{D}_{\text{val}}^{\text{clean}}$ but kept in the eval suite. ASDiv has a single monolithic validation set: there is no disjoint subset that can serve as held-out without leaking into test. We treat ASDiv as a *transfer* target: not contributing to m_b, but evaluated downstream. This is consistent with how Rho-1's reference model is trained on math without seeing ASDiv per se but is evaluated on it.

ICL exemplars are leave-one-out within each benchmark. For each held-out item v in benchmark b, we sample K other items as in-context exemplars. Sampling is subject-matched for MMLU, task-matched for BBH, and otherwise random within the benchmark. Where the source split permits, K follows the evaluation default, including three shots for BBH and eight for GSM8K. Because MMLU has only five development items per subject, its held-out prompts use one-shot leave-one-out ICL. This construction keeps the refresh prompts close to the evaluation format.

MathQA rationale filter. The crowdsourced MATHQA rationale field contains degenerate strings such as "`answer : a`." We require the final 150 characters to contain the gold-answer letter in a regex-matchable form, which removes approximately 15% of the training rationales. Without this filter, m_b shifts toward an answer-letter direction rather than multi-step reasoning; in an internal diagnostic, the average over the six non-ASDiv benchmarks decreased by approximately 0.4 points.

D TRAINING AND EVALUATION CONFIGURATION

This section specifies the data pool, evaluation suite, and training hyperparameters used in Section 5. All methods use the same training pool and evaluation protocol, and individual fine-tuning runs share the listed optimiser and batching settings. Method-specific reference preparation and repeated rescoring are described separately.

D.1 DATA POOL

The DS2-50k pool introduced in Section 5.1 contains 50,000 samples selected by an LLM-based quality-rating pipeline (Pang et al., 2024b). Table 5 gives its per-source composition and token-length statistics. We use the "Tulu" template (Wang et al., 2023) with `<|User|>` and `<|Assistant|>` role tags; lengths are measured with the LLaMA-3.2-3B tokenizer.

Table 5: DS2-50k composition. Setup matches Token Cleaning (Pang et al., 2025) for direct comparability.

Dataset	Source	Quality	#Samples	Prompt Len	Completion Len	Total Len
Open-Assistant 1	Human-generated	Both	2,418	45.4	364.1	409.5
Stanford Alpaca	Generated w/ Davinci-003	Normal	4,598	22.5	84.8	107.2
WizardLM	ChatGPT-generated	High	34,772	126.1	419.2	545.4
Dolly	Human-generated	Normal	1,567	119.1	133.7	252.9
Flan-v2	Human-generated	Normal	6,645	475.1	38.6	513.7

D.2 EVALUATION SUITE

All downstream evaluations use `lm-evaluation-harness` (Gao et al., 2023) with the official task definitions and default metrics. Table 6 lists the seven-benchmark math-and-reasoning suite reported in Table 1.

Table 6: The 7-bench math/reasoning evaluation suite. "Primary metric" is the metric reported in Table 1; all metrics are computed by `lm-evaluation-harness` on the full evaluation split (or a fixed limit for compute-bound benches).

Benchmark	Capability	Shots	Limit	Primary metric
MMLU (Hendrycks et al., 2020)	knowledge / reasoning	5	full	accuracy (`acc,none`)
BBH (Suzgun et al., 2022)	reasoning	3	40 / task	exact-match CoT (`get-answer`)
GSM8K (Cobbe et al., 2021)	math	8	full	exact-match CoT (`flexible-extract`)
MathQA (Amini et al., 2019)	math (MCQ)	5	full	accuracy norm
GSM-Plus-Mini (Li et al., 2024)	math	8	full	exact-match CoT
ASDiv-CoT-LLaMA (Miao et al., 2020)	math	5	full	exact-match CoT
Minerva-MATH (Lewkowycz et al., 2022)	math	4	100	exact-match

Why this suite. We focus on math and reasoning because the conditions of Proposition 1 are most plausible for benchmarks with coherent gradient directions (Appendix A). This suite therefore evaluates the intended regime of the direction-matching surrogate; it does not establish performance beyond that scope.

Under the ASDIV-COT-LLAMA answer-extraction protocol, the untuned base model has an anomalously low score, whereas the fine-tuned models produce compatible outputs. ASDiv is excluded from $\mathcal{D}_{\text{val}}$ because it has no clean disjoint split (Appendix C).

D.3 Training Hyperparameters

Table 7 lists the hyperparameters shared by the fine-tuning runs used for the main results. Reference construction and rescoring for the reference-based baselines are method-specific preparation steps outside these runs.

Table 7: Training hyperparameters for the main experiments (Section 5), shared across all fine-tuning runs.

Hyperparameter	Value
Optimiser	AdamW
Learning rate	1×10^{-4}, linear schedule, warmup ratio 0.03
Epochs	1
Sequence length	2048
Effective batch size	48
Precision	bfloat16
LoRA rank / α / dropout	64 / 16 / 0.1
LoRA targets	all attention and MLP projections
Selected proportion k	60%, per-response top-k% on the signed score
Selection warmup T_w	15% of training steps (full-token SFT)
Held-out refresh interval K_m	100 training steps
Held-out EMA decay α_m	0.7
Reference warmup (baselines)	first 10k samples, 1 epoch, same hyperparameters
Random seed	42 (single run)

Effective batch size. We match Token Cleaning's effective batch size of 48 (Pang et al., 2025). Its setup uses 8×L40S GPUs without accumulation, whereas ours uses 4×A5000 GPUs with gradient accumulation.

Held-out direction EMA. Appendix E specifies the direction-refresh schedule and EMA update. Score standardisation uses only the current batch and maintains no state across steps.

Reproducibility. Training scripts, evaluation scripts, and the held-out-set construction pipeline will be released upon publication. The DS2-50k data pool is publicly available (Pang et al., 2024b); all 7 evaluation benchmarks are accessible via the official `lm-evaluation-harness`.

E Implementation Details

Closed-form last-layer gradient. We evaluate the last-layer gradient in Eq. 4 using detached, 32-bit floating-point probabilities computed from the logits already produced by the supervised fine-tuning cross-entropy. The scoring computation therefore reuses the training forward pass and introduces neither an additional model forward pass nor an additional backward pass.

Batch standardisation and function-token gating. The aggregated alignment and the base-to-current loss difference are standardised separately over the trainable response-token positions of the current batch, with the alignment standardised after its softmax-weighted benchmark aggregation. No calibration statistics are maintained across training steps. In the implementation, the hidden-state boost is disabled for tokens in a fixed list of function words and punctuation; digits and mathematical operators are excluded from this list and therefore keep it.

Held-out refresh. The refresh schedule and its hyperparameters follow Section 4.2, Algorithm 1, and Appendix D. At each refresh, we update the cached directions as

$$m_b \leftarrow \alpha_m m_b + (1-\alpha_m) m_b^{\text{new}}, \qquad m_b^h \leftarrow \text{unit}\big(\alpha_m m_b^h + (1-\alpha_m) m_b^{h,\text{new}}\big), \tag{16}$$

so only the hidden-state direction is re-normalised after the update. The held-out construction enforces a 1024-token cap on each combined prompt and completion.