GRAIN: GROUP AGGREGATION VIA MIN-NORM OBJECTIVE FOR STABLE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Two training runs that differ only in the random seed can produce noticeably different models. The problem is long-standing, but it is most acute in the overparameterized regime of modern deep learning, where large models fine-tuned on limited data traverse flat loss landscapes with many near-equivalent minima, and it is most costly for large pretrained models (LPMs), whose training is expensive, whose downstream data is scarce, and for which repeated runs to average out variance are prohibitive. We trace a common source of this instability to the update rule itself: when the gradients of different groups of examples conflict in direction, their arithmetic mean can be small or even zero, stalling training while the loss is still high. We introduce **GRAIN** (**G**roup **A**ggregation via m**IN**-norm objective), a lightweight, optimizer-agnostic procedure that replaces the arithmetic mean, both within and across mini-batches, with the min-norm convex combination of group-wise gradients. The resulting update is guaranteed to conflict with no group gradient, so no group loss increases to first order. Under mild smoothness and absolute-continuity assumptions it differs almost surely from the mean, which yields a uniform-stability bound strictly tighter than SGD's. Empirically, across generation, classification, and regression at LPM scale, GRAIN is the only method in our suite with no collapsed run over 10 seeds, while improving mean performance and reducing run-to-run variance over a broad set of baselines, at no extra wall-clock or storage cost under data-parallel training.

1 INTRODUCTION

Stochastic optimization is at the heart of modern machine learning, but it comes with a cost: two runs of the same algorithm on the same data, with the same training configuration, can produce noticeably different models. Weight initialization, data ordering, mini-batch sampling, dropout masks, and even hardware-level non-determinism all feed into this variance (Picard, 2021; Summers & Dinneen, 2021; Madhyastha & Jain, 2019; Bethard, 2022). In practice, these sources of randomness, except hardware non-determinism, are typically governed by a single hyperparameter, the random seed, so that the seed alone can determine whether an otherwise identical training configuration succeeds or fails. The phenomenon is ubiquitous: it appears in CNNs trained from scratch on imbalanced or noisy data, in regression with pretrained encoders fine-tuned on small targets, and most prominently in the fine-tuning of large pretrained models (Dodge et al., 2020; Mosbach et al., 2020; Zhang et al., 2020; Bui et al., 2025). Fig. 1 illustrates the magnitude of the problem: fixing all hyperparameters and varying only the seed produces over 50 percentage points (pp) gap between the best and worst seeds on PubMedQA, isolated catastrophic failures on SuperGLUE BoolQ, and smaller but persistent variance on noisy-label image classification using non-LPM models. Controlling these randomness sources, whether individually or jointly, is non-trivial, and even when it succeeds it leaves the underlying cause untouched. We therefore target the mechanism that turns seed randomness into divergent outcomes: *the optimization process*, which prior work has treated as a tuning problem rather than an aggregation problem.

Why instability is more severe in the overparameterized regime. Classical learning theory studies the under-parameterized regime, where the training loss has a unique (or nearly unique) minimizer and the randomness of stochastic gradient descent (SGD) averages away. Modern deep learning operates in a fundamentally different regime: models are massively over-parameterized

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.

with respect to the amount of available training data, the loss landscape admits a continuum of near-zero loss solutions (Zhang et al., 2021), and small perturbations in the optimization trajectory can route training toward very different minima with very different generalization behavior (Mosbach et al., 2020; Dodge et al., 2020). Overparameterization is not the cause of randomness, but it is the *amplifier* that turns everyday stochasticity into macroscopic test-accuracy swings.

Why the LPM era makes this urgent. Three properties of the current LPM era conspire to make instability particularly costly: *(i) Models are large*: A single fine-tuning run of a 7B or 14B model can take hours on multiple GPUs; repeating it many times to average away seed noise is often infeasible in practice. *(ii) Downstream data is scarce*: Fine-tuning sets for domain-specific tasks are typically small, keeping the system deep in the overparameterized regime where instability is worst. *(iii) Deployment stakes are high*: Models fine-tuned for medical, legal, or safety-critical applications are expected to behave consistently; run-to-run variance is a reliability liability, not just a research nuisance. A training algorithm that reduces seed-induced variance *without* multiplying compute or storage is therefore valuable well beyond a single domain.

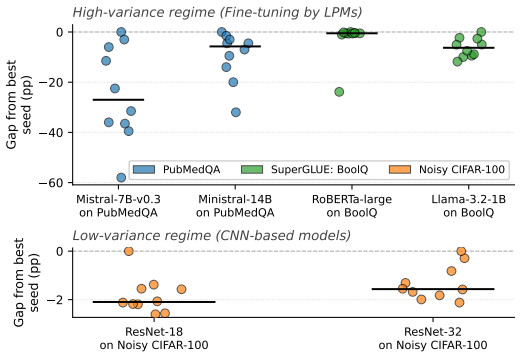


Figure 1: Seed-induced accuracy variance across six (model, task) configurations, each trained for 10 random seeds with identical hyperparameters. Each point is plotted as the gap below that configuration's best performance .

The limits of existing remedies. *Ensembles and weight averaging* (SnapshotEnsemble Wang et al. (2020; 2023), Model Soups Wortsman et al. (2022), Model Stocks Jang et al. (2024), Stochastic Weight Averaging (Izmailov et al., 2018; Gao et al., 2022; Madhyastha & Jain, 2019; Nishida et al., 2025)) are effective but scale training cost or storage with ensemble size, making them prohibitive at scale. *Data-centric* remedies such as larger datasets Dodge et al. (2020) or longer schedules Mosbach et al. (2020) are often impractical and still leave substantial variance. *Noise injection* (LNSR (Hua et al., 2021), NoisyTune (Wu et al., 2022)), *sharpness-aware methods* (SAM Foret et al. (2020), ASAM Kwon et al. (2021), Tilted-SAM Li et al. (2024)), and *regularization schemes* (label smoothing Müller et al. (2019), Mixout Lee et al. (2020), Dropout Srivastava et al. (2014), R-Drop Wu et al. (2021)) all target generalization rather than the optimization process itself. *Gradient-conflict resolvers* from multi-task learning (PCGrad (Yu et al., 2020), CAGrad (Liu et al., 2021), GradDrop (Chen et al., 2020), MGDA (Sener & Koltun, 2018)) require a separate backward pass per "task" and detect conflict via cosine similarity, which fails when gradients are orthogonal or small in magnitude. What is missing is a method that (i) *resolves gradient conflict inside the update rule with a guarantee that the update opposes none of the gradients it aggregates, rather than suppressing the randomness that triggers it*, (ii) *adds negligible compute or storage*, and (iii) *comes with a stability guarantee*.

Our approach. We argue that the arithmetic mean used in SGD, both within a mini-batch (intra-batch) and across mini-batches (inter-batch), is what exposes training to seed-induced instability: when the gradients of different groups of examples conflict in direction, their mean can be small or even zero and the update stalls, a failure that learning-rate tuning cannot repair (Section 2). We therefore replace the mean with the min-norm convex combination of the group gradients (Section 3), whose direction by construction decreases every group loss and whose norm is never larger than the mean's (Section 4). The update costs one backward pass plus a small quadratic program in the number of groups; under data-parallel training the per-device gradients already form the groups, so wall-clock cost is unchanged.

Contributions. (i) We introduce GRAIN, an optimizer-agnostic training procedure that aggregates group-wise gradients via a min-norm convex combination, replacing both intra-batch and inter-batch mean operations. (ii) We prove, for any smooth model trained on data with a continuous density, that GRAIN conflicts with no group, converges at rate $\mathcal{O}(1/T)$, and admits a uniform-stability bound

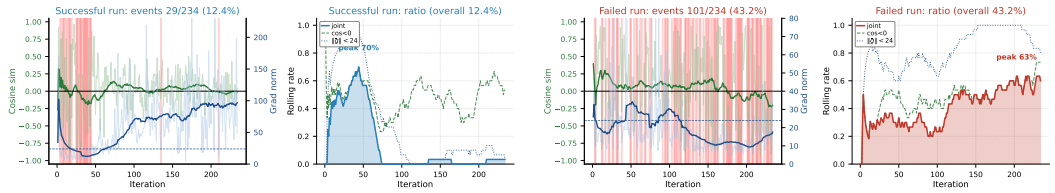


Figure 2: Gradient cancellation across training (`RoBERTa-large` on RTE), where a cancellation event is an iteration with $\cos(g_i, g_j) < 0$ AND mean aggregated gradient norm $\|\bar{g}\| = \frac{1}{2}(\|g_i + g_j\|)$ below the 25th percentile of the successful run's norms (≈ 23.9). Each run shows per-iteration cosine and norm with red-shaded events (panels 1, 3) and the rolling-30 fraction satisfying the joint condition, $\cos < 0$ alone, and $\|\bar{g}\| < 23.9$ alone (panels 2, 4). The failed run (accuracy 52.71%) exhibits 43.2% cancellation versus 12.4% for the successful run (accuracy 84.47%), with a sustained late-training rate above 50%.

strictly tighter than SGD's (Section 4). (iii) We validate GRAIN empirically across generation, sequence and image classification, and regression at LPM scale, reporting consistent improvements in mean performance and run-to-run variance over a broad suite of baselines.

2 PHENOMENON

Before motivating GRAIN's design, we formalize the failure modes we target, illustrate it on a representative case (Section 5 shows it across tasks), and identify a common underlying mechanism: *group-level gradient cancellation*.

Detecting per-example gradient conflict in high-dimensional parameter space is infeasible. We instead measure conflict at the *group-level*, both **(1)** *intra-batch* (between groups within a single mini-batch) and **(2)** *inter-batch* (between consecutive mini-batches), by inspecting two quantities along the training trajectory: **aggregated gradient norm** and **between-group cosine similarity** of the group-wise gradients. A small net gradient combined with negative between-group cosine similarity is the signature of *gradient cancellation*: the update stalls while the loss is still high because the group gradients oppose one another, in contrast to convergence, where the update stops because the loss is already low and no group has a descent direction left.

Motivation. Fig. 2 makes the cancellation pattern concrete. We fine-tune `RoBERTa-large` on RTE for two runs differing only in random seed; within each mini-batch, examples are partitioned uniformly at random into two groups g_i and g_j. Cancellation events ($\cos(g_i, g_j) < 0$ AND small aggregated gradient norm) occupy 43.2% of training iterations in the failed run versus 12.4% in the successful one, with the failed run's rolling rate climbing past 50% in late training while the successful run's decays after step ~ 70. Cancellation is specific neither to LPMs nor to any particular way of partitioning a mini-batch. Any grouping by class, by loss magnitude, by residual sign, or by arbitrarily index can produce group gradients that oppose one another, and the arithmetic mean used by SGD averages these opposing signals indiscriminately. Similar evidence can be observed in the across-mini-batch gradient conflicts shown in Fig. 8 in Appendix E.5. This gradient cancellation can not be fixed by tuning learning rate since the group gradients still conflict while a larger step in a conflicting direction leads to exploding gradients and a smaller step will make the cancellation worse (Figure 5c). This motivates a principled alternative that is *task-agnostic* and *partition-agnostic*: rather than averaging group gradients, find **the convex combination of group gradients with the smallest norm, which by construction conflicts with no group**. The problem was never the magnitude but the direction: the min-norm point is the one direction guaranteed to be descent for every group (Theorem 4.2), and its small norm is what yields the tighter stability bound (Theorem 4.4). This min-norm objective is the core of GRAIN. We use cosine similarity here only as a post-hoc diagnostic to visualize conflict; GRAIN itself never thresholds on it, which is what separates it from detection-based resolvers such as PCGrad.

3 PROPOSED SOLUTION: GRAIN

Let $\mathcal{D}$ be a data distribution over $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, $f_\theta : \mathcal{X} \to \mathcal{Y}$ a model with parameters $\theta \in \Theta \subseteq \mathbb{R}^d$, and $\mathcal{L}(\theta, z)$ a differentiable per-example loss. **At iteration t** we sample a mini-batch $\mathcal{B}_t$ of size B i.i.d. from $\mathcal{D}$. Standard mini-batch SGD updates $\theta_{t+1} = \theta_t - \eta \cdot \frac{1}{B} \sum_{z_i \in \mathcal{B}_t} \nabla_\theta \mathcal{L}(\theta_t, z_i)$.

GRAIN update. We partition $\mathcal{B}_t$ into m disjoint groups $\mathcal{B}_t = \bigcup_{i=1}^m \mathcal{B}_{t,i}$, with $\mathcal{B}_{t,i} \cap \mathcal{B}_{t,j} = \emptyset$ for $i \neq j$. Let $g_i = \frac{1}{|\mathcal{B}_{t,i}|} \sum_{z \in \mathcal{B}_{t,i}} \nabla_\theta \mathcal{L}(\theta_t, z)$ be the group gradient. Define the convex hull of group gradients as:

$$\mathcal{G} = \Big\{ g \in \mathbb{R}^d \;\Big|\; g = \sum_{i=1}^m \alpha_i g_i,\ \alpha \in \Delta^{m-1} \Big\}, \quad \Delta^{m-1} = \Big\{ \alpha \in \mathbb{R}^m_{\geq 0} : \sum_i \alpha_i = 1 \Big\}.$$

SGD corresponds to $\alpha = \frac{1}{m}\mathbf{1}$. GRAIN instead solves the *min-norm* problem Sener & Koltun (2018):

$$\alpha_t^\star \;=\; \arg\min_{\alpha \in \Delta^{m-1}} \Big\| \sum_{i=1}^m \alpha_i g_i \Big\|^2, \qquad \bar{g}_t \;=\; \sum_{i=1}^m \alpha_{t,i}^\star \, g_i, \tag{1}$$

and performs the update:

$$\theta_{t+1} \;=\; \theta_t \;-\; \eta \, \bar{g}_t. \tag{2}$$

The problem in equation 1 is a convex QP in m variables; we solve it via the Frank–Wolfe algorithm of Sener & Koltun (2018), which converges in $\mathcal{O}(m)$ inner iterations. For the small m we use in practice ($m \in \{2, 3, 4\}$), solving equation 1 takes microseconds relative to one forward/backward of an LPM. The full algorithm is given in Appendix B.

Two levels of aggregation. The grouping can be applied both *within* a mini-batch (**intra-batch**) and *across* mini-batches (**inter-batch**). In the intra-batch setting, groups are formed inside $\mathcal{B}_t$, for example by splitting $\mathcal{B}_t$ into m sub-batches (our default). In the inter-batch setting, the "groups" are k consecutive mini-batch gradients $\{G_t, G_{t-1}, \ldots, G_{t-k+1}\}$, and GRAIN aggregates them with a min-norm combination. Inter-batch GRAIN requires only the gradients already computed in the previous $k-1$ steps; no additional backward pass is needed. Section 5.3 ablates both choices.

Optimizer-agnostic. The min-norm aggregation is independent of the choice of first-order optimizer. To use GRAIN with AdamW, Adafactor, or any other adaptive method, replace the batch-mean gradient passed to the optimizer with $\bar{g}_t$ from equation 1 and run the optimizer's usual update; the only change in the training loop is which scalar combination of group gradients is fed to the optimizer.

Implementation & computational cost. On distributed LLM training with *multi-GPU*, GRAIN composes naturally with data parallelism: each GPU's local-shard gradient is one group g_i, an all-reduce gathers them on rank 0 which solves the min-norm QP and broadcasts $\alpha_t^\star$, and each device forms $\bar{g}_t$ as a weighted sum locally. The communication payload matches vanilla data-parallel training (gradient tensors plus one length-m vector broadcast), and the QP solve is microseconds, so per-iteration wall-clock cost is effectively unchanged for m up to the number of GPUs. For the *single-GPU* setting, the backward computational cost scale linearly with the number of group m due to we have to divide the mini-batch into groups for separate gradient calculation.

4 THEORETICAL ANALYSIS

We state the four core theoretical properties of GRAIN. Throughout, we use η for the learning rate, L_s for the smoothness constant of $\mathcal{L}$ (the Lipschitz constant of $\nabla\mathcal{L}$), and H for the Lipschitz constant of $\mathcal{L}$ in θ (so $\|\nabla\mathcal{L}\| \leq H$) and we use a **full-batch training** setting; this formulation can be readily generalized to mini-batch training by replacing the corresponding empirical quantities with their expectations. Proofs are deferred to Appendix C.

Definition 4.1 (Instability). *Given a stochastic training algorithm $\mathcal{A}$ and a test point z, the (uniform) instability of $\mathcal{A}$ at z is:*

$$\mathrm{Ins}_{z,\mathcal{A}} = \mathbb{E}\big[\,|\mathcal{L}(\theta, z) - \mathcal{L}(\theta', z)|\,\big], \tag{3}$$

where θ and θ' are parameters returned by two independent runs of $\mathcal{A}$ from different seeds (initializations and/or data orderings) and the expectation is over the seed randomness.

Theorem 4.2 (Zero group-gradient conflict). *Let $\bar{g}$ be the GRAIN update defined in equation 1. Then for every group gradient g_i, $\langle g_i, \bar{g}\rangle \geq \|\bar{g}\|^2 \geq 0$. In particular, taking a step $-\eta\bar{g}$ does not increase any group loss.*

Theorem 4.2 follows directly from the KKT conditions of the min-norm problem and distinguishes GRAIN from PCGrad-style projections which only zero conflict after detection via negative cosine similarity.

Theorem 4.3 (Convergence). *Assume each per-example loss is L_s-smooth and bounded below by $\mathcal{L}^\star$, and $0 < \eta < 1/L_s$. Let $\{\theta_t\}$ be generated by the GRAIN update equation 2. Then:*

$$\min_{0\leq t<T} \|\bar{g}_t\|^2 \;\leq\; \frac{2(\mathcal{L}(\theta_0) - \mathcal{L}^\star)}{(\eta - L_s\eta^2/2)\,T} \;=\; \mathcal{O}(1/T), \tag{4}$$

so $\bar{g}_t \to 0$. When all groups are i.i.d. from $\mathcal{D}$, this implies $\nabla\mathcal{L}(\theta_t) \to 0$, i.e. θ_t converges to a stationary point with a finite update steps.

The strictness in our stability bound rests on a key technical fact: under mild assumptions (smooth activations, continuous data density, and a generic-position assumption on the parametric family), the min-norm solution $\alpha^\star$ differs almost surely from the uniform weight $\frac{1}{m}\mathbf{1}$, so $\|\bar{g}\| < \|\frac{1}{m}\sum_i g_i\|$ almost surely. The full statement and proof are deferred to Theorem C.7 in Appendix C.5.

Theorem 4.4 (Strictly tighter instability bounds). *Assume $\mathcal{L}$ is H-Lipschitz in θ and the conditions of Theorem C.7 (Appendix C.5) hold. The instability upper bound of GRAIN is strictly lower than that of SGD while using the same learning rate schedule $\{\eta_t\}_{t=0}^{T-1}$, more specifically:*

$$\mathrm{Ins}_{z\sim\mathcal{D},\mathit{GRAIN}} < 2H^2\sum_{i=0}^{T-1}\eta_t + \|\theta_0 - \theta_0'\|, \quad \mathrm{Ins}_{z\sim\mathcal{D},\mathit{SGD}} \leq 2H^2\sum_{i=0}^{T-1}\eta_t + \|\theta_0 - \theta_0'\|, \tag{5}$$

where GRAIN and SGD are examined from the same starting points of two different initialization θ_0 and θ_0' from different seeds.

Theorem 4.4 bounds an expected point-wise loss difference. Smaller worst-case-allowed instability for GRAIN is a plausible—but not sufficient—condition for lower run-to-run variability in final accuracy, since the latter also depends on optimization path, data order, and initialization. We therefore treat the accuracy STD in Section 5 as empirical evidence consistent with the theory, not as a quantity the theory predicts.

Definition 4.5 (Sharpness). *The ρ-sharpness of the loss function $\mathcal{L}$ at θ is:*

$$S_\rho(\theta) := \mathit{max}_{\|\epsilon\|\leq\rho}\mathcal{L}(\theta+\epsilon) - \mathcal{L}(\theta). \tag{6}$$

Theorem 4.6 (Loss sharpness bound). *GRAIN loss function can be rewrite as a weighted aggregation of all group losses: $\mathcal{L}(\theta) = \sum_{i=1}^m \alpha^\star l_i(\theta)$. Assuming that each loss function of individual group $\{l_i\}_{i=1}^m$ is L-smooth, GRAIN loss sharpness at θ is bounded:*

$$S_\rho^{GRAIN}(\theta) := \mathit{max}_{\|\epsilon\|\leq\rho}\mathcal{L}(\theta+\epsilon) - \mathcal{L}(\theta) \leq \rho\|\bar{g}\| + \frac{\rho^2 L}{2}. \tag{7}$$

If assumptions in Theorem C.7 hold i.e. we have $\|\bar{g}\| < \|g\|$ hence the sharpness bound for GRAIN strictly tighter than SGD ones. The empirical evidence of Theorem 4.6 is provided in Figure 4 in Appendix E.6.

5 Empirical experiments

5.1 Experiment design

We conduct experiment on various kinds of task including **text generation**, **classification** and **regression**. Detailed experimental setups and additional results can be found in Apx E.1

For generative task: we focus on question answering task on two domains mathematic and healthcare by using GSM8K and PubmedQA for this sotisphicated task we leverage LLMs including two model families Qwen and Mistral ranging from 7B to 14B including `Qwen2-7B-base` Yang et al. (2024a), `Qwen2.5-14B-base` Yang et al. (2024b) and Mistral family: `Mistral-7B-v0.3`, `Ministral-3-14B-Base-2512` Liu et al. (2026).

For classification task: both sequence and image classification are used for the evaluation for this task, we use simple pretrained models: `Roberta-large` and `Llama-3.2-1B` for *sequence classification* and evaluate all methods on BoolQ, RTE from SuperGLUE Wang et al. (2019); and MRPC from GLUE Wang et al. (2018). For *image classification* we adopt `ViT-base` as the backbone model and evaluate all training techniques on CIFAR dataset under **train-time distribution shift**, where the training data is perturbed by randomness. Models are then evaluated on the same clean, balanced test set, so performance variance reflects the algorithm's stability rather than evaluation noise. In this study, we consider two types of distribution shift: class imbalance Cui et al. (2019); Azizzadenesheli et al. (2019) and label noise Jiang et al. (2020) (used for low-variance regime analysis in Section 5.4).

Baselines. We compare GRAIN against (1) fully fine-tuning (**FFT**) or LoRA Hu et al. (2022) (LLMs above 1B) fine-tuning (**LoRA**); (2) FFT/LoRA with focal loss Lin et al. (2017) (**FocalLoss**); *noise injection approaches (for pretrained models)*: (3) **NoisyTune** Wu et al. (2022); *gradient conflict resolver*: we apply (4) **PCGrad** Yu et al. (2020) on gradients according to each individual group using the random same grouping strategy; (5) stochastic weight averaging (**SWA**) Izmailov et al. (2018); (6) **SAM** Foret et al. (2020). We report mean performance (*Mean*) and standard deviation (*STD*) over 10 arbitrarily chosen random seeds. Full hyperparameters, dataset statistics, and baseline implementations are in Appendix E. Detailed grouping settings of GRAIN are listed in Table 6.

5.2 Main results

Since our aim is to mitigate seed-induced instability and, where possible, eliminate the failed runs it produces, rather than to maximize average accuracy, we judge a method by three criteria: (1) no collapsed run across the 10 seeds; (2) low seed-to-seed variance; (3) competitive, if not the highest, mean accuracy. We call a run *collapsed* when its test accuracy falls to the majority-class or chance level, and mark any cell containing at least one such run with * in Tables 1, 2, 3.

Generative task results. Table 1 reports the results of generative tasks. GRAIN is **the only method that avoids collapsed failure on every (task, model) run with the best or the second best mean accuracies** (*-free across both tasks; LoRA, NoisyTune, and SWA each show * on at least two PubMedQA cells, including the dramatic LoRA failure on `Mistral-7B` at $46.05_{\pm 19.00}$). On PubMedQA, where seed instability is most severe due to the small labeled split, GRAIN attains the best Mean on `Qwen-7B` (74.15, $+1.55$ over LoRA) and `Qwen-14B` (76.58, $+2.00$ over LoRA), and stabilizes the catastrophic `Mistral-7B` run from $46.05_{\pm 19.00}$ to $67.60_{\pm 0.61}$. On the more stable GSM8K benchmark, where every method runs without collapse, GRAIN achieves the best or tied-best Mean on every model with the lowest STD across the board, showing that GRAIN preserves accuracy and tightens variance even when the baseline is already stable.

Classification task results. Table 2 reports sequence classification results; Table 3 reports image classification results. The same pattern holds in both: GRAIN is **the only method without a single collapsed failure with the best or the second best mean accuracies**, while every baseline produces at least three starred entries. On RoBERTa-large, FFT and noise-injection baselines collapse on every sequence task (STD $\approx$ 5–16); GRAIN cuts STD by an order of magnitude (e.g., RTE: $82.59_{\pm 1.24}$ vs. FFT $76.17_{\pm 15.44}$) and posts the best clean Mean on BoolQ ($85.04_{\pm 0.38}$). On `Llama-3.2-1B`, FocalLoss exhibits universal training failures (every entry starred), while GRAIN is the only method that beats FFT on BoolQ (84.32 vs. 73.98) with substantially improved stability. On imbalanced CIFAR (Table 3), GRAIN achieves the best mean accuracy on all eight (dataset $\times$ imbalance type $\times$ ratio) cells with consistently the lowest or near-lowest STD; the gain is largest on the more challenging step-imbalance settings, where SAM and FocalLoss exhibit super high STDs (e.g., FocalLoss STD $=$ 14.22 on CIFAR-10 ST-100, SAM STD $=$ 11.55 on CIFAR-10 LT-50) while GRAIN stays under 1.0 throughout.

Robustness does not come at the cost of accuracy. GRAIN attains the highest mean in 5 of 8 generative configurations, 5 of 8 sequence-classification configurations, and all 8 imbalanced-

Table 1: Generative LLM fine-tuning using LoRA. Mean ± STD of exact-match accuracy (%) across 10 seeds. Best per column bold; second-best underlined. * indicates method includes at least 1 collapsed failure.

Method	PubMedQA				GSM8K			
	Qwen-7B	Qwen-14B	Mistral-7B	Ministral-14B	Qwen-7B	Qwen-14B	Mistral-7B	Ministral-14B
LoRA	72.60±0.52	74.58±2.08	*46.05±19.00	*67.90±9.94	72.87±0.71	82.94±0.35	47.95±0.74	79.24±0.50
+ FocalLoss	71.62±0.62	73.80±1.03	*20.60±8.40	74.66±0.76	73.98±0.65	82.18±0.65	47.53±0.75	76.42±1.51
+ NoisyTune	70.05±0.55	74.20±0.67	*64.86±7.44	76.70±0.82	74.43±0.45	84.46±0.42	48.20±0.78	78.77±0.96
+ SWA	68.15±1.25	73.65±0.85	*67.55±5.76	75.95±0.37	73.90±0.29	85.45±0.51	48.65±0.88	78.77±0.72
+ SAM	70.95±1.09	75.65±1.20	66.95±2.71	**76.80±1.32**	72.88±0.69	85.45±0.93	46.49±0.89	79.63±0.80
+ PCGrad	66.40±0.81	76.10±0.52	**69.90±1.10**	*73.15±3.28	74.63±0.54	**86.40±0.55**	48.14±0.48	79.12±0.73
+ GRAIN	**74.15±0.58**	**76.58±1.16**	67.60±0.61	75.20±0.59	**74.77±0.34**	85.53±0.15	**49.67±0.48**	**79.85±0.54**

Table 2: Sequence classification and regression performance across different finetuning strategies.

Method	RTE	MRPC	BoolQ	STS-B	RTE	MRPC	BoolQ	STS-B
	RoBERTa-large				Llama-3.2-1B			
FFT	*76.17±15.44	*84.90±5.33	*83.20±7.39	**91.86±0.51**	79.49±1.83	**86.69±1.05**	*73.98±3.85	90.12±0.36
+ FocalLoss	77.29±1.70	*86.52±6.50	81.22±0.46	88.61±2.76	*49.39±2.97	*68.38±0.00	*62.17±0.00	83.66±2.21
+ NoisyTune	*70.65±15.53	*86.91±6.62	*71.04±11.46	91.38±0.70	74.69±2.92	84.76±2.08	67.64±1.01	83.97±1.81
+ SWA	*78.45±13.61	**88.19±1.21**	*80.36±9.61	91.18±1.06	76.17±1.47	67.29±1.64	82.62±3.79	85.14±0.33
+ SAM	*69.27±10.92	*74.83±8.41	*80.61±8.34	87.58±2.21	75.27±3.14	82.49±1.84	70.07±1.21	89.21±0.31
+ PCGrad	*81.05±10.08	*86.67±6.52	*83.22±7.40	91.04±0.56	73.88±2.95	82.84±3.39	67.77±1.21	90.11±0.21
+ GRAIN	**82.59±1.24**	87.84±0.74	**85.04±0.38**	91.56±0.36	**82.22±1.12**	85.21±1.39	**84.32±0.68**	**90.78±0.41**

Table 3: Image classification on imbalanced CIFAR using ViT-base. LT = long-tailed imbalance, ST = step imbalance; the number after each denotes the imbalance ratio.

Method	CIFAR-10				CIFAR-100			
	LT-100	LT-50	ST-100	ST-50	LT-100	LT-50	ST-100	ST-50
ViT (FFT)	88.47±1.77	96.46±0.44	83.27±2.13	96.09±0.51	69.55±2.71	85.53±0.48	56.73±1.58	84.30±0.48
+ FocalLoss	89.32±0.68	95.29±3.58	*70.31±14.22	97.15±0.57	*64.68±10.70	87.20±2.21	57.19±2.49	*86.60±4.17
+ NoisyTune	87.88±1.67	*88.94±9.39	*68.86±15.40	96.80±0.24	*68.05±3.75	87.89±0.24	57.17±1.11	87.89±0.24
+ SWA	*39.29±4.29	*90.58±5.31	*45.66±4.80	*71.37±16.83	*51.14±9.60	*66.96±13.46	*35.62±18.60	*73.08±15.65
+ SAM	*88.07±9.41	*92.15±11.55	82.77±1.12	96.57±0.22	*63.55±11.08	87.81±0.91	58.50±1.60	87.84±0.02
+ PCGrad	86.63±2.44	96.85±0.19	*82.05±3.57	96.84±0.15	68.13±1.47	87.02±0.37	50.01±1.18	87.24±0.37
+ GRAIN	**89.83±0.88**	**96.97±0.13**	**85.17±0.90**	**97.56±0.13**	**70.59±0.38**	**88.87±0.09**	**60.17±0.43**	**87.91±0.09**

CIFAR configurations. No baseline is best in more than 2 of 8 in any task family. Where GRAIN is not best, the winner has higher variance in all but one case. Two mechanisms explain this. On LPM tasks the seed outcome is bimodal: healthy runs reach near-zero training loss, so the mean is set by how many runs fall into the collapsed tail. Theorem 4.4 tightens the stability bound and thus rules out the worst trajectories (Hardt et al., 2016); removing the tail raises the mean.

Why baselines fail unevenly across task categories. We offer the following explanations. They rest on how each baseline handles gradient conflict by design; our experiments do not test them directly. Among the baselines, only PCGrad acts on gradient conflict at all, and it does so only partially: it modifies the update when a pairwise cosine is negative, otherwise reduces to the mean, and bounds neither the direction nor the norm of the final update, so it inherits SGD's instability outside the pairs it corrects. The remaining baselines leave the directions of the group gradients untouched, so conflicting groups remain in conflict. FocalLoss reweights examples by confidence, which changes how much each example contributes but not where the group gradients point. SAM perturbs along the same conflicting mean gradient; moreover, the collapsed constant-prediction solution is itself a flat region, so the sharpness penalty is small there and SAM can settle into it rather than escape it. NoisyTune injects isotropic noise that is blind to the directions responsible for divergence. SWA averages nearby checkpoints, which, once training has entered the collapsed basin, all sit in that basin. In addition, on imbalanced CIFAR, FFT never collapses, yet the baselines fail unevenly across the eight configurations: *distribution shift exposes baseline-specific weaknesses that a clean training distribution conceals*. GRAIN is the only method with no collapsed run in the entire suite. We attribute this to a min-norm aggregation that resolves the conflict itself, is applied at every step at both intra- and inter-batch levels, and does not depend on detecting any particular conflict signature.

5.3 Ablation Study and Sensitivity Analysis

We analyze the contribution of each level at which the Min-Norm solver is applied in GRAIN: **(1)** *a global-level* (inter-batch), where the min-norm objective is applied across consecutive mini-batches, and **(2)** *a local-level* (intra-batch), where the min-norm objective is applied to gradients computed over groups obtained by partitioning each mini-batch. In addition, we study the effectiveness and robustness of GRAIN with respect to **(3)** the number of consecutive mini-batches (k) and **(4)** the number of groups used to partition each mini-batch (m). Concretely, we use `Qwen2-7B` as the backbone model and evaluate variants of GRAIN on PubMedQA. The results are shown in Figure 3.

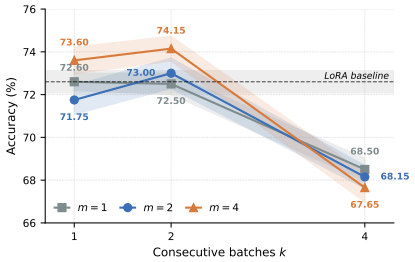


Figure 3: Sensitivity analysis on PubmedQA using Qwen2-7B as the backbone model.

Ablation. The LoRA baseline reaches 72.60 ± 0.52 on PubMedQA, which corresponds to $(m = 1, k = 1)$ in our factorial. Each level alone provides limited benefit. Pure global aggregation $(m = 1, k = 2)$ yields 72.50, statistically indistinguishable from baseline. Pure local aggregation $(k = 1)$ with $m = 2$ underperforms baseline at 71.75, while $m = 4$ surpasses it at 73.60, indicating that finer partitioning is required for the local-level to contribute on its own. The combination $(m = 4, k = 2)$ reaches 74.15, the best cell observed and the only configuration that clearly beats baseline. The two levels are therefore complementary rather than redundant. The local-level solver attacks intra-batch gradient conflict, while the global-level solver damps cross-batch trajectory noise that the group-level cannot see; neither mechanism alone fully captures the variance GRAIN reduces, but their composition does.

Sensitivity. *Number of groups* m. The theory motivates "the larger the better" for m: in the extreme case where each group contains a single example, the min-norm solver resolves every individual gradient conflict within a mini-batch, recovering the strongest form of the no-conflict guarantee in Theorem 4.2. The result is consistent with this at $k \leq 2$, where accuracy is monotone in m ($m = 1 < m = 2 < m = 4$ at $k = 2$). The ordering inverts at $k = 4$, but only because all three m values break at that setting; the inversion reflects a failure of the global-level rather than a problem with finer partitioning. *Number of consecutive mini-batches* k. Performance improves from $k = 1$ to $k = 2$ for the partitioned settings ($m \in \{2, 4\}$) but degrades sharply at $k = 4$ across all m, losing roughly 4–5 points relative to baseline. The story for k is asymmetric. *Recent* cross-batch gradient conflicts are genuinely harmful and worth resolving, since the current update is directly impacted by the previous one; *distant* cross-batch gradients are largely irrelevant, since the parameters have already moved through several updates and the relationship resembles a short-memory Markov chain in which only the recent past carries information about the present.

Practical takeaway. Set m as large as compute allows and $k = 2$. With our multi-GPU implementation m scales naturally up to the number of devices at no extra wall-clock cost, so the configuration that maximizes m subject to the GPU budget is the one we recommend.

5.4 Low-Variance Regime: Non-LPM Models

While the most dramatic seed effects appear in fine-tuning of large pretrained models, seed-induced variance is also present when training non-LPM models from scratch or under noisy supervision, although the variance is quite small in magnitude. To verify that GRAIN's benefits extend to this regime, we evaluate it on noisy-label and imbalanced image classification with `ResNet-18` and `ResNet-32` on CIFAR-10 and CIFAR-100 across four label-noise ratios $\{0.2, 0.4, 0.6, 0.8\}$ detailed results are shown in Table 4. Across most tasks, GRAIN achieves higher accuracy with lower variance. GRAIN improves accuracy on 29 of 32 configurations and lower variance on 22 of 32. Even in this regime, where each configuration's seed-to-seed spread is on the order of 1 percentage point, GRAIN's group gradient modulation produces consistently improvements, indicating that the gradient-cancellation phenomenon addressed by GRAIN is not unique to LPM fine-tuning, but rather a general characteristic of mini-batch training.

In this experiment, we also provide empirical evidence supporting Theorem 4.6. Specifically, `ResNet` without the residual connections trained with GRAIN exhibits a smoother loss landscape

Table 4: `Resnet-{18, 32}` performance using normal training baseline and + GRAIN (m=2, k=1) on CIFAR with **noisy label** and **imbalance** settings.

	Noisy Label				Imbalance			
Ratio	0.2	0.4	0.6	0.8	LT-100	LT-50	ST-100	ST-50
Method	**CIFAR-10**							
`Resnet-18`	86.47±0.28	81.91±0.54	73.27±0.65	43.51±0.83	**67.97±0.78**	93.01±0.16	61.15±0.71	93.03±0.14
+ GRAIN	**87.06±0.34**	**82.59±0.29**	**74.36±0.79**	**44.19±0.49**	67.53±0.21	**93.03±0.08**	**61.62±0.56**	**93.11±0.13**
`Resnet-32`	86.77±0.27	82.51±0.73	74.08±0.94	40.73±2.21	65.80±1.04	**92.93±0.44**	62.13±0.93	**93.08±0.16**
+ GRAIN	**87.47±0.25**	**83.20±0.48**	**75.08±0.87**	**41.67±2.17**	**67.22±0.99**	92.71±0.31	**63.69±0.16**	93.00±0.06
Method	**CIFAR-100**							
`Resnet-18`	63.66±0.40	50.80±0.74	39.41±0.57	19.42±0.76	36.80±0.24	71.98±0.41	41.06±0.22	71.67±0.09
+ GRAIN	**64.12±0.33**	**52.47±0.72**	**40.99±0.74**	**20.46±0.53**	**37.29±0.78**	**72.10±0.51**	**41.16±0.33**	**71.80±0.19**
`Resnet-32`	65.50±0.44	52.62±0.75	40.65±1.42	19.29±0.68	35.31±0.95	71.96±0.37	40.88±0.09	71.78±0.46
+ GRAIN	**65.76±0.50**	**53.77±0.42**	**42.33±0.66**	**20.12±0.46**	**35.96±1.07**	**72.29±0.30**	**41.10±0.40**	**72.07±0.05**

than the original SGD. Detailed experimental settings and additional results are provided in Appendix E.6.

6 LIMITATIONS

GRAIN addresses seed-induced variance through gradient aggregation; it does not address other sources of instability such as prompt formatting (He et al., 2024) or in-context example selection (Gupta et al., 2023). Our stability proof assumes smooth activations, an absolutely continuous data density, and a generic-position condition (Assumption C.9) that holds for almost every parameter setting of standard overparameterized networks; extending it to piecewise-linear networks such as ReLU requires additional work. The min-norm QP is cheap for the group counts we use ($m \leq 4$), but its $\mathcal{O}(m^2)$ cost makes very large m impractical without a sparse-support approximation, and on a single GPU the backward cost grows linearly with m because each group needs its own gradient. The empirical instability (the standard deviation of the test metric across seeds) we report is not the quantity bounded in Theorem 4.4, which is the expected loss difference between two runs at a single test point; a tighter bound on the latter does not formally imply a smaller value of the former, so we present the theory as motivation and the seed variance as evidence consistent with it. Finally, our sensitivity analysis covers $m \in \{2, 4\}$ and $k \in \{1, 2, 4\}$; behavior at larger m or k, and on architectures we did not test, notably Mamba and other state-space models, is left to future work.

7 CONCLUSION

We introduced GRAIN, an optimizer-agnostic training procedure that replaces the arithmetic mean used by mini-batch optimization with a min-norm convex combination of group-wise gradients, applied at both intra-batch and inter-batch levels. GRAIN carries three guarantees that hold for any smooth model trained on data with a continuous density: zero group-wise gradient conflict, $\mathcal{O}(1/T)$ convergence, and a strictly tighter uniform-stability bound than SGD. Empirically across generative LLM fine-tuning, sequence and image classification, and regression at LPM scale, GRAIN is the only method in our suite that avoids collapsed failure entirely while consistently improving mean performance and reducing run-to-run variance, at a per-iteration cost indistinguishable from standard data-parallel training.

AI USE STATEMENT

(This section is **required** and does not count toward the page limit.)

In this work, we used generative AI tools for polishing the draft. We did not use generative AI tools for idea generation or theoretical analysis and proof. The remaining required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for coding implementation. We have carefully reviewed all AI-assisted work and revised all sentences and code modified or generated with the assistance of AI. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Kamyar Azizzadenesheli, Anqi Liu, Fanny Yang, and Animashree Anandkumar. Regularized learning for domain adaptation under label shifts. *arXiv preprint arXiv:1903.09734*, 2019.

Steven Bethard. We need to talk about random seeds. *arXiv preprint arXiv:2210.13393*, 2022.

Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural networks*, 106:249–259, 2018.

Nghia Bui, Guergana Savova, and Lijing Wang. Assessing the macro and micro effects of random seeds on fine-tuning large language models, 2025. URL https://arxiv.org/abs/2503.07329.

Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in neural information processing systems*, 32, 2019.

Zhao Chen, Jiquan Ngiam, Yanping Huang, Thang Luong, Henrik Kretzschmar, Yuning Chai, and Dragomir Anguelov. Just pick a sign: Optimizing deep multitask models with gradient sign dropout. *Advances in Neural Information Processing Systems*, 33:2039–2050, 2020.

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.

Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9268–9277, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. 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. 4171–4186, 2019.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*, 2020.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

Yingbo Gao, Christian Herold, Zijian Yang, and Hermann Ney. Revisiting checkpoint averaging for neural machine translation. *arXiv preprint arXiv:2210.11803*, 2022.

Shivanshu Gupta, Matt Gardner, and Sameer Singh. Coverage-based example selection for in-context learning. *arXiv preprint arXiv:2305.14907*, 2023.

Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *International conference on machine learning*, pp. 1225–1234. PMLR, 2016.

Jia He, Mukund Rungta, David Koleczek, Arshdeep Sekhon, Franklin X Wang, and Sadid Hasan. Does prompt formatting have any impact on llm performance? *arXiv preprint arXiv:2411.10541*, 2024.

Jiangpeng He. Gradient reweighting: Towards imbalanced class-incremental learning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16668–16677. IEEE, 2024.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Morris W Hirsch. *Differential topology*. Springer Science & Business Media, 2012.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

Hang Hua, Xingjian Li, Dejing Dou, Chengzhong Xu, and Jiebo Luo. Noise stability regularization for improving bert fine-tuning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3229–3241, 2021.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

Dong-Hwan Jang, Sangdoo Yun, and Dongyoon Han. Model stock: All we need is just a few fine-tuned models. In *European Conference on Computer Vision*, pp. 207–223. Springer, 2024.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL `https://arxiv.org/abs/2310.06825`.

Lu Jiang, Di Huang, Mason Liu, and Weilong Yang. Beyond synthetic noise: Deep learning on controlled noisy labels. In *International conference on machine learning*, pp. 4804–4815. PMLR, 2020.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. Pubmedqa: A dataset for biomedical research question answering. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 2567–2577, 2019.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Jungmin Kwon, Jeongseop Kim, Hyunseo Park, and In Kwon Choi. Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. In *International conference on machine learning*, pp. 5905–5914. PMLR, 2021.

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. Mixout: Effective regularization to finetune large-scale pretrained language models. In *International Conference on Learning Representations (ICLR)*, 2020.

Buyu Li, Yu Liu, and Xiaogang Wang. Gradient harmonized single-stage detector. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 8577–8584, 2019.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31, 2018.

Tian Li, Tianyi Zhou, and Jeffrey A Bilmes. Tilted sharpness-aware minimization. *arXiv preprint arXiv:2410.22656*, 2024.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

Alexander H. Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, Alexandre Sablayrolles, Amélie Héliou, Amos You, Andy Ehrenberg, Andy Lo, Anton Eliseev, Antonia Calvi, Avinash Sooriyarachchi, Baptiste Bout, Baptiste Rozière, Baudouin De Monicault, Clémence Lanfranchi, Corentin Barreau, Cyprien Courtot, Daniele Grattarola, Darius Dabert, Diego de las Casas, Elliot Chane-Sane, Faruk Ahmed, Gabrielle Berrada, Gaëtan Ecrepont, Gauthier Guinet, Georgii Novikov, Guillaume Kunsch, Guillaume Lample, Guillaume Martin, Gunshi Gupta, Jan Ludziejewski, Jason Rute, Joachim Studnia, Jonas Amar, Joséphine Delas, Josselin Somerville Roberts, Karmesh Yadav, Khyathi Chandu, Kush Jain, Laurence Aitchison, Laurent Fainsin, Léonard Blier, Lingxiao Zhao, Louis Martin, Lucile Saulnier, Luyu Gao, Maarten Buyl, Margaret Jennings, Marie Pellat, Mark Prins, Mathieu Poirée, Mathilde Guillaumin, Matthieu Dinot, Matthieu Futeral, Maxime Darrin, Maximilian Augustin, Mia Chiquier, Michel Schimpf, Nathan Grinsztajn, Neha Gupta, Nikhil Raghuraman, Olivier Bousquet, Olivier Duchenne, Patricia Wang, Patrick von Platen, Paul Jacob, Paul Wambergue, Paula Kurylowicz, Pavankumar Reddy Muddireddy, Philomène Chagniot, Pierre Stock, Pravesh Agrawal, Quentin Torroba, Romain Sauvestre, Roman Soletskyi, Rupert Menneer, Sagar Vaze, Samuel Barry, Sanchit Gandhi, Siddhant Waghjale, Siddharth Gandhi, Soham Ghosh, Srijan Mishra, Sumukh Aithal, Szymon Antoniak, Teven Le Scao, Théo Cachet, Theo Simon Sorg, Thibaut Lavril, Thiziri Nait Saada, Thomas Chabal, Thomas Foubert, Thomas Robert, Thomas Wang, Tim Lawson, Tom Bewley, Tom Bewley, Tom Edwards, Umar Jamil, Umberto Tomasini, Valeriia Nemychnikova, Van Phung, Vincent Maladière, Virgile Richard, Wassim Bouaziz, Wen-Ding Li, William Marshall, Xinghui Li, Xinyu Yang, Yassine El Ouahidi, Yihan Wang, Yunhao Tang, and Zaccharie Ramzi. Ministral 3, 2026. URL https://arxiv.org/abs/2601.08584.

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. *Advances in Neural Information Processing Systems*, 34:18878–18890, 2021.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Pranava Swaroop Madhyastha and Rishabh Jain. On model stability as a function of random seed. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pp. 929–939, 2019.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. On the stability of fine-tuning bert: Misconceptions. *Explanations, and Strong Baselines. arXiv*, 2020.

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? *Advances in neural information processing systems*, 32, 2019.

Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

Yuto Nishida, Masaru Isonuma, and Yusuke Oda. Instability in downstream task performance during LLM pretraining. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 22883–22895, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi: 10.18653/v1/2025.findings-emnlp.1246. URL https://aclanthology.org/2025.findings-emnlp.1246/.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

David Picard. Torch. manual_seed (3407) is all you need: On the influence of random seeds in deep learning architectures for computer vision. *arXiv preprint arXiv:2109.08203*, 2021.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. *Advances in neural information processing systems*, 31, 2018.

Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 761–769, 2016.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Cecilia Summers and Michael J Dinneen. Nondeterminism and instability in neural network optimization. In *International Conference on Machine Learning*, pp. 9913–9922. PMLR, 2021.

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.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP workshop BlackboxNLP: Analyzing and interpreting neural networks for NLP*, pp. 353–355, 2018.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32, 2019.

Lijing Wang, Dipanjan Ghosh, Maria Gonzalez Diaz, Ahmed Farahat, Mahbubul Alam, Chetan Gupta, Jiangzhuo Chen, and Madhav Marathe. Wisdom of the ensemble: Improving consistency of deep learning models. *Advances in Neural Information Processing Systems*, 33:19750–19761, 2020.

Lijing Wang, Yingya Li, Timothy Miller, Steven Bethard, and Guergana Savova. Two-stage fine-tuning for improved bias and variance for large pretrained language models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15746–15761, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.877. URL https://aclanthology.org/2023.acl-long.877/.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pp. 23965–23998. PMLR, 2022.

Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. Noisytune: A little noise can help you finetune pretrained language models better. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 680–685, 2022.

Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, Tie-Yan Liu, et al. R-drop: Regularized dropout for neural networks. *Advances in neural information processing systems*, 34:10890–10905, 2021.

An Yang et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024a.

An Yang et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024b.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in neural information processing systems*, 33: 5824–5836, 2020.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. Revisiting few-sample bert fine-tuning. *arXiv preprint arXiv:2006.05987*, 2020.

A Related Work

Learning instability across settings. Run-to-run variance has been documented across the ML stack. Madhyastha & Jain (2019) and Bethard (2022) study seed sensitivity in classical NLP pipelines. Picard (2021) and Summers & Dinneen (2021) show substantial variance in from-scratch training of computer-vision models attributable to seed choice and hardware non-determinism. In the fine-tuning of LPMs, Devlin et al. (2019); Dodge et al. (2020); Lee et al. (2020) attribute instability to catastrophic forgetting and small fine-tuning datasets, while Mosbach et al. (2020) point to optimization difficulties and vanishing gradients and recommend small learning rates, bias-corrected Adam, and long training schedules; we follow these guidelines in our FFT baseline and still observe high variance. Wang et al. (2023) formally decompose LPM fine-tuning variance into sampling and optimization components and show that the optimization component grows with overparameterization. The common thread is that stochasticity in the optimization trajectory matters more when the loss landscape admits many near-equivalent solutions which is the overparameterized regime.

Ensembles and weight averaging. Ensembles (Wang et al., 2020; 2023) are the most reliable mitigation but scale training cost with ensemble size. Model Soups (Wortsman et al., 2022) and Model Stock (Jang et al., 2024) average weights of multiple fine-tuned models and belong to the same cost category. Stochastic weight averaging (Izmailov et al., 2018; Gao et al., 2022; Madhyastha & Jain, 2019; Nishida et al., 2025) avoids training multiple models but requires extra storage for checkpoints and can fail when the local basin itself is degenerate.

Noise injection and sharpness-aware methods. LNSR (Hua et al., 2021) and NoisyTune (Wu et al., 2022) inject noise into weights or representations. Sharpness-aware methods (SAM (Foret et al., 2020), ASAM (Kwon et al., 2021), Tilted-SAM (Li et al., 2024)) minimize loss under worst-case parameter perturbations and require an extra forward–backward pass per step. Regularization methods such as label smoothing (Müller et al., 2019), Mixout (Lee et al., 2020), Dropout (Srivastava et al., 2014), and R-Drop (Wu et al., 2021) operate on weights or outputs rather than on gradient aggregation.

Gradient conflict resolvers. PCGrad (Yu et al., 2020), CAGrad (Liu et al., 2021), GradDrop (Chen et al., 2020), and MGDA (Sener & Koltun, 2018) address task-level gradient conflict in multitask learning. We adapt these methods to the single-task setting by grouping examples within a mini-batch. They act only on gradients with significantly negative cosine similarity, miss orthogonal or small-magnitude conflicts, and require a backward pass per group. GHM (Li et al., 2019) and OHEM (Shrivastava et al., 2016) reweight gradients by difficulty but do not target seed-induced instability. In contrast, GRAIN operates on any group partition, uses a single backward pass, and admits a clean stability guarantee.

B Algorithm

Algorithm 1 summarizes intra-batch GRAIN. The inter-batch variant is obtained by replacing the group construction in line 3 with a ring buffer of past k gradients.

Algorithm 1 GRAIN (intra-batch variant)

Require: initial parameters θ_0; learning rate η; group count m; iterations T

1: **for** $t = 0, 1, \ldots, T-1$ **do**
2: sample mini-batch $\mathcal{B}_t \sim \mathcal{D}^B$
3: partition $\mathcal{B}_t$ into m disjoint groups $\{\mathcal{B}_{t,i}\}_{i=1}^m$
4: compute group gradients $g_i = \frac{1}{|\mathcal{B}_{t,i}|} \sum_{z \in \mathcal{B}_{t,i}} \nabla_\theta \mathcal{L}(\theta_t, z)$
5: solve $\alpha_t^\star \leftarrow \arg\min_{\alpha \in \Delta^{m-1}} \| \sum_i \alpha g_i \|^2$ (Frank–Wolfe)
6: $\bar{g}_t \leftarrow \sum_i \alpha_{t,i}^\star g_i$
7: $\theta_{t+1} \leftarrow \theta_t - \eta\, \bar{g}_t$
8: **end for**
9: **return** θ_T

C LEMMAS, THEOREMS, AND PROOFS

We use η_t for the learning rate at step t, L_s for the smoothness constant (Lipschitz constant of $\nabla\mathcal{L}$), and H for the Lipschitz constant of $\mathcal{L}$ in θ (so $\|\nabla_\theta\mathcal{L}\| \le H$).

C.1 PRELIMINARIES

Definition C.1 (L_s-smooth function)**.** *$f : \mathbb{R}^d \to \mathbb{R}$ is L_s-smooth if ∇f is L_s-Lipschitz, equivalently $|f(y) - f(x) - \langle \nabla f(x), y - x\rangle| \le \frac{L_s}{2}\|y - x\|^2$.*

Definition C.2 (H-Lipschitz function)**.** *$f : \Theta \times \mathcal{Z} \to \mathbb{R}$ is H-Lipschitz in θ if $\|\nabla_\theta f(\theta, z)\| \le H$ for all θ, z, which implies $|f(\theta, z) - f(\theta', z)| \le H\|\theta - \theta'\|$.*

Lemma C.3 (Descent lemma (Nesterov, 2013))**.** *For an L_s-smooth f and any x, y, $f(y) \le f(x) + \langle \nabla f(x), y - x\rangle + \frac{L_s}{2}\|y - x\|^2$.*

Lemma C.4 (Min-norm KKT)**.** *Let $\mathcal{G} = \{g = \sum_i \alpha_i g_i : \alpha \in \Delta^{m-1}\}$ and $g^\star = \sum_i \alpha_i^\star g_i$ where $\alpha^\star = \arg\min_{\alpha\in\Delta^{m-1}} \|g\|^2$. Then for all $g \in \mathcal{G}$, $\langle g, g^\star\rangle \ge \|g^\star\|^2$. In particular, $\langle g_i, g^\star\rangle \ge \|g^\star\|^2$ for every group gradient g_i.*

Proof. Convexity of $\mathcal{G}$ gives $g^\star + \epsilon\Delta \in \mathcal{G}$ for $\Delta = g - g^\star$ and $\epsilon \in [0, 1]$. Optimality of $g^\star$ implies $\|g^\star + \epsilon\Delta\|^2 \ge \|g^\star\|^2$, i.e. $2\epsilon\langle\Delta, g^\star\rangle + \epsilon^2\|\Delta\|^2 \ge 0$. Letting $\epsilon \to 0^+$ yields $\langle\Delta, g^\star\rangle \ge 0$, hence $\langle g, g^\star\rangle \ge \|g^\star\|^2$. The statement for g_i follows from $g_i \in \mathcal{G}$ (set $\alpha_j = \delta_{ij}$). □

C.2 PROOF OF THEOREM 4.2

Lemma C.4 applied to $g^\star = \bar{g}$ gives $\langle g_i, \bar{g}\rangle \ge \|\bar{g}\|^2 \ge 0$ for every i. □

C.3 PROOF OF THEOREM 4.3

Proof. Apply Lemma C.3 at $\theta_{t+1} = \theta_t - \eta\bar{g}_t$:

$$\mathcal{L}(\theta_{t+1}) \le \mathcal{L}(\theta_t) - \eta\langle\nabla\mathcal{L}(\theta_t), \bar{g}_t\rangle + \tfrac{L_s\eta^2}{2}\|\bar{g}_t\|^2.$$

Writing $\nabla\mathcal{L}(\theta_t) = \frac{1}{m}\sum_i g_i$ and using Theorem 4.2, $\langle\nabla\mathcal{L}(\theta_t), \bar{g}_t\rangle = \frac{1}{m}\sum_i\langle g_i, \bar{g}_t\rangle \ge \|\bar{g}_t\|^2$. Hence

$$\mathcal{L}(\theta_{t+1}) \le \mathcal{L}(\theta_t) - \big(\eta - \tfrac{L_s\eta^2}{2}\big)\|\bar{g}_t\|^2.$$

For $0 < \eta < 1/L_s$ the coefficient is positive. Telescoping from $t = 0$ to $T - 1$:

$$\sum_{t=0}^{T-1}\|\bar{g}_t\|^2 \le \frac{2(\mathcal{L}(\theta_0) - \mathcal{L}^\star)}{\eta - L_s\eta^2/2}, \quad \min_{0\le t<T}\|\bar{g}_t\|^2 \le \frac{2(\mathcal{L}(\theta_0) - \mathcal{L}^\star)}{(\eta - L_s\eta^2/2)\,T}.$$

For i.i.d. groups, $\mathbb{E}[\bar{g}_t] = \mathbb{E}[\nabla\mathcal{L}(\theta_t)]$ (since each $\lambda_t^\star$ is invariant under joint relabeling of groups), so $\bar{g}_t \to 0$ implies $\nabla\mathcal{L}(\theta_t) \to 0$. □

C.4 TOOLS FOR MEASURE-THEORETIC STATEMENTS

Notation: $\lambda^m(f)$ denotes that f has m-dimensional Lebesgue measure zero.

Lemma C.5 (C^1 image of compact cube has measure zero)**.** *Let $K \subset \mathbb{R}^k$ be a compact cube and $f : K \to \mathbb{R}^m$ a C^1 map with $k < m$. Then the Lebesgue $\lambda^m(f(K)) = 0$.*

Proof. Df is bounded on K by some L_f, so f is Lipschitz with constant L_f. Fix $N \in \mathbb{N}$ and cover K by N^k sub-cubes of side $\delta = \mathrm{diam}(K)/N$. Each sub-cube Q_i has diameter $\sqrt{k}\,\delta$, so $f(Q_i)$ is contained in a ball of radius $L_f\sqrt{k}\,\delta$, which fits inside an m-dimensional cube of side $2L_f\sqrt{k}\,\delta$. Summing, $\lambda^m(f(K)) \le N^k(2L_f\sqrt{k}\,\delta)^m = C \cdot N^{k-m}$ for $C = (2L_f\sqrt{k}\,\mathrm{diam}(K))^m$. Since $k < m$, $N^{k-m} \to 0$ as $N \to \infty$, so the Lebesgue $\lambda^m(f(K)) = 0$. □

Theorem C.6 (Submanifolds of positive codimension have measure zero)**.** *Let $M \subset \mathbb{R}^m$ be a k-dimensional C^1 submanifold with $k < m$. Then the Lebesgue $\lambda^m(M) = 0$.*

Proof. M admits a countable atlas $\{f_i : K_i \to \mathbb{R}^m\}_{i\geq 1}$ with $K_i \subset \mathbb{R}^k$ compact (using second-countability of M and compact exhaustion of chart domains). By Lemma C.5, $\lambda^m(f_i(K_i)) = 0$. Countable sub-additivity gives the Lebesgue $\lambda^m(M) \leq \sum_i \lambda^m(f_i(K_i)) = 0$. □

C.5 Min-norm differs from the mean almost surely

Theorem C.7 (Min-norm differs from the mean, a.s.)**.** *Let f_θ be a model with smooth activations and let $z_1, \ldots, z_m$ be i.i.d. samples from a distribution with continuous positive density p on $\mathcal{X}$. Define $g_i = \nabla_\theta \mathcal{L}(\theta, z_i)$ and $\alpha^\star = \arg\min_{\alpha\in\Delta^{m-1}} \|\sum_i \alpha_i g_i\|^2$. Under Assumption C.9 below (satisfied for almost every θ in the model parameter space), for almost every θ,*

$$\Pr\big(\alpha^\star = \tfrac{1}{m}\mathbf{1}\big) = 0,$$

where the probability is taken over the data sampling.

Corollary C.8 (Strict gradient-norm separation)**.** *Under the conditions of Theorem C.7, $\|\bar{g}\| < \|\frac{1}{m}\sum_{i=1}^m g_i\|$ almost surely.*

Proof of Corollary C.8. Min-norm optimality of $\alpha^\star$ gives $\|\bar{g}\| \leq \|\frac{1}{m}\sum_i g_i\|$ deterministically. The inequality is strict whenever $\alpha^\star \neq \frac{1}{m}\mathbf{1}$, which by Theorem C.7 happens almost surely. □

Proof of Theorem C.7. The proof requires care: a direct application of Sard's theorem to Ψ (the constraint map) shows that the set of *critical values* of Ψ in $\mathbb{R}^{m-1}$ has measure zero, which does *not* imply that the specific value $0 \in \mathbb{R}^{m-1}$ is a regular value. The correct tool is the parametric transversality theorem.

Setup. Fix a model architecture and let $g_i(\theta, z) = \nabla_\theta \mathcal{L}(\theta, z_i) \in \mathbb{R}^d$ be the per-example gradient at parameter $\theta \in \Theta$ and data point $z_i \in \mathcal{X}$. Smooth activations make g_i a C^∞ function of (θ, z_i). Define $\bar{g}_{\text{mean}} = \frac{1}{m}\sum_i g_i$ and $\Phi_i(\theta; z_1, \ldots, z_m) = \langle g_i, \bar{g}_{\text{mean}}\rangle - \|\bar{g}_{\text{mean}}\|^2$ for $i = 1, \ldots, m$. Since $\sum_i \Phi_i \equiv 0$ identically, only $m-1$ are independent; collect them into $\Psi_\theta : \mathcal{X}^m \to \mathbb{R}^{m-1}$, $\Psi_\theta = (\Phi_1, \ldots, \Phi_{m-1})$. By KKT, the equal-weight event is exactly $\{\lambda^\star = \frac{1}{m}\mathbf{1}\} = \Psi_\theta^{-1}(0)$. $\Psi_\theta \in C^\infty(\mathcal{X}^m; \mathbb{R}^{m-1})$ for each fixed θ.

Joint map. Define

$$F : \Theta \times \mathcal{X}^m \to \mathbb{R}^{m-1}, \qquad F(\theta, z_1, \ldots, z_m) = \Psi_\theta(z_1, \ldots, z_m).$$

F is C^∞ jointly in (θ, z) by the smoothness assumption.

Assumption C.9 (Generic position of the parametric family)**.** *F is a submersion at every point of $F^{-1}(0)$, i.e. the differential dF has full rank $m-1$ everywhere on $F^{-1}(0)$. Equivalently, 0 is a regular value of the joint map F.*

This is a mild genericity condition: when the parametric family is overparameterized ($\dim\Theta \gg d$) and has continuously varying gradients, dF generically has full row rank. It is not a strong restriction in practice; in particular, for almost every randomly initialized smooth network, Assumption C.9 holds. We treat it as standing throughout this section.

Parametric transversality. We invoke the following standard result.

Theorem C.10 (Parametric transversality; Hirsch 2012, Thm. 3.2.7)**.** *Let Θ, $\mathcal{X}^m$, and $\mathbb{R}^{m-1}$ be smooth manifolds and $F : \Theta \times \mathcal{X}^m \to \mathbb{R}^{m-1}$ a C^∞ map for which $y \in \mathbb{R}^{m-1}$ is a regular value. Then for almost every $\theta \in \Theta$ (in any measure absolutely continuous w.r.t. Lebesgue measure on Θ), y is a regular value of the restricted map $\Psi_\theta := F(\theta, \cdot) : \mathcal{X}^m \to \mathbb{R}^{m-1}$.*

Combining Assumption C.9 (which makes 0 a regular value of F) with Theorem C.10 yields: for almost every $\theta \in \Theta$, 0 is a regular value of Ψ_θ.

Conclusion. Fix any such θ in the full-measure subset of Θ furnished by parametric transversality. By the preimage theorem, $\Psi_\theta^{-1}(0)$ is a smooth submanifold of $\mathcal{X}^m$ of dimension $mD-(m-1)$, where $D=\dim\mathcal{X}$. Since $mD-(m-1)<mD$, Theorem C.6 gives $\lambda^{mD}(\Psi_\theta^{-1}(0))=0$.

The data law $z_1,\dots,z_m \overset{\text{i.i.d.}}{\sim} p$ has joint density $\prod_i p(z_i)$, which is absolutely continuous w.r.t. Lebesgue measure on $\mathcal{X}^m$. Absolutely continuous measures assign zero probability to Lebesgue-null sets, so

$$\Pr\big(\lambda^\star=\tfrac{1}{m}\mathbf{1}\big)=\Pr\big((z_1,\dots,z_m)\in\Psi_\theta^{-1}(0)\big)=\int_{\Psi_\theta^{-1}(0)}\prod_i p(z_i)\,dz_1\cdots dz_m=0.$$

This proves Theorem C.7; the gradient-norm strict inequality $\|\bar g\|<\|\frac{1}{m}\sum_i g_i\|$ a.s. follows as Corollary C.8. □

C.6 Strictly tighter expansion bound

Lemma C.11 (Strict step-size separation)**.** *Under the assumptions of Theorem C.7, with probability one over the data sampling,*

$$\|\theta-U_{\mathrm{GRAIN}}(\theta)\|\;<\;\|\theta-U_{\mathrm{SGD}}(\theta)\|\;\le\;\eta H,$$

where $U_{\mathrm{GRAIN}}(\theta)=\theta-\eta\bar g$ *and* $U_{\mathrm{SGD}}(\theta)=\theta-\eta\,\frac{1}{m}\sum_i g_i$.

Proof. $\|\theta-U_{\mathrm{GRAIN}}(\theta)\|=\eta\|\bar g\|$, $\|\theta-U_{\mathrm{SGD}}(\theta)\|=\eta\|\frac{1}{m}\sum_i g_i\|$. The strict inequality $\|\bar g\|<\|\frac{1}{m}\sum_i g_i\|$ a.s. is Corollary C.8. The upper bound ηH follows from $\|g_i\|\le H$ for every i and convexity of the norm. □

C.7 Proof of Theorem 4.4

Proof. Fix a data sampling sequence $\mathcal{B}_0,\dots,\mathcal{B}_{T-1}$ shared between the two runs (this is the standard data-coupled Hardt–Recht–Singer setup). Let $\Delta_t=\|\theta_t-\theta'_t\|$ for two GRAIN runs from initializations $\theta_0\neq\theta'_0$, and $\Delta'_t=\|\psi_t-\psi'_t\|$ for the corresponding SGD runs. For SGD,

$$\Delta'_{t+1}=\|\psi_t-\psi'_t-\eta_t(g_{t,\mathrm{SGD}}-g'_{t,\mathrm{SGD}})\|\le\Delta'_t+\eta_t(\|g_{t,\mathrm{SGD}}\|+\|g'_{t,\mathrm{SGD}}\|)\le\Delta'_t+2\eta_t H.$$

Iterating: $\Delta'_T\le\|\theta_0-\theta'_0\|+2H\sum_{t=0}^{T-1}\eta_t$. For GRAIN, the same triangle-inequality decomposition gives

$$\Delta_{t+1}\le\Delta_t+\eta_t(\|\bar g_t\|+\|\bar g'_t\|).$$

By Lemma C.11, $\|\bar g_t\|<\|\frac{1}{m}\sum_i g_{t,i}\|\le H$ a.s. at every step, and likewise for the primed run. Hence $\Delta_{t+1}<\Delta_t+2\eta_t H$ a.s. at every step, and

$$\Delta_T<\|\theta_0-\theta'_0\|+2H\sum_{t=0}^{T-1}\eta_t\quad\text{a.s.}$$

Taking expectations preserves the strict inequality because the event of equality has probability zero. By H-Lipschitzness of $\mathcal{L}$, $\mathrm{Ins}_{z,\mathcal{A}}\le H\,\mathbb{E}[\Delta_T]$. Combining, equation 5 follows. □

C.8 Proof for Theorem 4.6

Proof. We have $S_\rho^{GRAIN}=\max_{\|\epsilon\|\le\rho}\mathcal{L}(\theta+\epsilon)-\mathcal{L}(\theta)$. Since $\{l_i\}_{i=1}^m$ is L-smooth hence $\mathcal{L}(\theta)$ is L-smooth:

$$\begin{aligned}
&l_i(\theta+\epsilon)\le l_i(\theta)+{g_i}^\top\epsilon+\frac{L}{2}\|\epsilon\|^2,\\
\Rightarrow&\sum_{i=1}^m\alpha^\star l_i(\theta+\epsilon)\le\sum_{i=1}^m\alpha^\star l_i(\theta)+(\sum_{i=1}^m\alpha^\star g_i)^\top\epsilon+\frac{L}{2}\|\epsilon\|^2,\\
\Rightarrow&\mathcal{L}(\theta+\epsilon)\le\mathcal{L}(\theta)+\bar g^\top\epsilon+\frac{L}{2}\|\epsilon\|^2,
\end{aligned}\tag{8}$$

hence:

$$\max_{\|\epsilon\|\leq\rho}\mathcal{L}(\theta+\epsilon)-\mathcal{L}(\theta)\leq\max_{\|\epsilon\|\leq\rho}[\bar{g}^\top\epsilon+\frac{L}{2}\|\epsilon\|^2]. \tag{9}$$

Cauchy-Schwarz inequality gives:

$$\bar{g}^\top\epsilon\leq\|\bar{g}\|\|\epsilon\|\leq\rho\|\bar{g}\|, \tag{10}$$

we also have:

$$\frac{L}{2}\|\epsilon\|^2\leq\frac{L}{2}\rho^2, \tag{11}$$

thus:

$$S_\rho^{GRAIN}(\theta)=\max_{\|\epsilon\|\leq\rho}\mathcal{L}(\theta+\epsilon)-\mathcal{L}(\theta)\leq\rho\|\bar{g}\|+\frac{\rho^2L}{2}. \tag{12}$$

□

Using the same proof framework as above we also obtain:

$$S_\rho^{SGD}(\theta)\leq\rho\|g\|+\frac{\rho^2L}{2}, \tag{13}$$

where $g=\frac{1}{m}\sum_{i=1}^{m}\nabla l_i(\theta)$.

D Implementation

Assume the mini-batch size is B In practical implementation, for a single GPU training, we leverage the gradient accumulation and set it to m, and our mini-batch size is B/m hence the operation is exactly the same as describe in the method section 3.

E Detailed Experiment Setup

In this section we detail the settings for our experiments.

E.1 Settings & Implementations

We arbitrarily choose 10 random seeds (42, 52, 62, 72, 82, 92, 102, 112, 122, 132) to obtain the mean and variance performance. All single runs are conducted on 2 NVIDIA-A100-80GB GPUs.

Baselines:

- For **PCGrad** Yu et al. (2020) we customize it for our task where we consider each group of training example in every iteration as an individual task in multi-task learning context and apply PCGrad learning procedure.
- For **NoisyTune** Wu et al. (2022), we add uniform noise $U(a,b)$, where λ controls the noise intensity. For the GLUE dataset, we set $\lambda=0.15$, following Wu et al. (2022). For other task we use the same λ setting.
- **SWA** Izmailov et al. (2018) maintains an average of model parameters over previous checkpoints. We tune the averaging window size in the range $3,5$ and report the best result for each task.
- Although the original implementation of **Focal Loss** Lin et al. (2017) suggests default hyperparameters of $\alpha=0.25$ and $\gamma=2$, the modified loss for a single data point z_i is defined as: $\mathcal{L}(z_i)=-\alpha_t(1-p_t)^\gamma\log(p_t)$ where p_t denotes the predicted probability of the ground-truth class. These configurations frequently induced training divergence or collapsed states in our preliminary evaluations. Consequently, we performed an extensive hyperparameter search for α and γ tailored to each task and pretrained architecture. The results reported herein represent the optimal performance achieved after this exhaustive tuning process.

- **SAM** Foret et al. (2020) add an adversarial gradient to minimize the generalization gap between the training loss and the testing one, in our experiment we set $\rho = 0.05$ as mentioned in the report.

Implementation: all approaches are optimized using the AdamW Loshchilov & Hutter (2017) optimizer, with the learning rate tuned over the range $5\text{E}{-6}, 1\text{E}{-5}, 2\text{E}{-5}, 3\text{E}{-5}, 1\text{E}{-4}, 3\text{E}{-4}$. We also experimented with a smaller learning rate of $1\text{E}{-6}$; however, collapsed states were still observed, while convergence was significantly slower as shown in Fig. 5c.

E.2 Generative Tasks

Models. We use four open-weight LLMs spanning two model families and two parameter scales: `Qwen2-7B-base` (Yang et al., 2024a) and `Qwen2.5-14B-base` (Yang et al., 2024b) from the Qwen family, and `Mistral-7B-v0.3` (Jiang et al., 2023) and `Ministral-3-14B-Base-2512` (Liu et al., 2026) from the Mistral family. All four are pretrained base checkpoints; we deliberately avoid instruction-tuned variants to keep the loss landscape comparable across models and to isolate seed-induced training instability from instruction-tuning artifacts.

Datasets. We evaluate on two generative reasoning benchmarks. **GSM8K** (Cobbe et al., 2021) contains 8.5K grade-school math word problems with chain-of-thought solutions; we use the standard 7.5K-train / 1.3K-test split. **PubMedQA** (Jin et al., 2019) is a biomedical QA dataset where the model answers *yes/no/maybe* given a research abstract and question; we use the expert-labeled partition (PQA-L, 1K examples) and split it 80%-20% into training and test sets. Both datasets exercise long-form generation conditioned on a structured input; together they cover quantitative reasoning (GSM8K) and qualitative biomedical inference (PubMedQA).

Fine-tuning protocol. We fine-tune all models with LoRA (Hu et al., 2022) on a single $8\times$ A100 (80GB) node. The hyperparameter settings are shown in Table 5.

Metrics. We evaluate in the zero-shot setting using **exact-match accuracy** as the primary metric. For PubMedQA, a prediction is correct if the generated answer normalizes to one of {`yes, no, maybe`} and matches the gold label; for GSM8K, the final numeric answer (extracted from the model's output via the standard regex) must match the gold answer.

E.3 Sequence Classification

Models: We adopt widely used models for sequence classification, covering two architectural paradigms: the encoder-only `RoBERTa-Large` Liu et al. (2019) and the decoder-only `LLaMA-3.2-1B` Touvron et al. (2023).

Datasets: BoolQ, RTE from SuperGLUE Wang et al. (2019); and MRPC from GLUE Wang et al. (2018).

We finetune pretrained language model with different type of architecture including encoder only: `RoBERTa-large`, decoder only: `Llama-3.2-1B`. To ensure a proper baseline implementation, we refer to the configurations in Liu et al. (2019); Hu et al. (2022) and replicate the state-of-the-art (SOTA) scores reported using `RoBERTa-large` with FFT. For `Llama-3.2-1B`, no established reference performance was available in the literature; we therefore fine-tune it using our own settings. Although our implementation may not achieve SOTA performance, it establishes a consistent basis for comparison across different methods.

E.4 Image classification

Models: We use `ResNet-18` and `ResNet-32` He et al. (2016) to evaluate GRAIN under label noise. For class imbalance experiments, we adopt `ViT-base-patch16-224` Dosovitskiy et al. (2021) as the backbone.

Datasets. CIFAR-10 and CIFAR 100 Krizhevsky et al. (2009) contain $50{,}000$ training images and $10{,}000$ test images of size 32×32, with 10 and 100 classes respectively. The test set is left untouched in both shift scenarios.

Settings: Following Cao et al. (2019); He (2024), we created imbalanced CIFAR-100 with two imbalance ratios (100:1 and 50:1) and two imbalance patterns: long-tailed (exponential decay) Cui et al. (2019) and step imbalance Azizzadenesheli et al. (2019), resulting in four distinct datasets. With noisy label settings Jiang et al. (2020) we randomly flip the label of a data point with ratios of $\{0.2, 0.4, 0.6, 0.8\}$.

Image classification under distribution shift. We use image classification to test GRAIN under *train-test distribution shift*, a setting known to amplify run-to-run variance in computer vision and one that places GRAIN's variance-reduction claim under stress. We construct shift in two complementary ways: (i) *class-imbalanced training* (varying class frequencies) and (ii) *noisy-label training* (corrupted supervision). In both cases the test set is held fixed at the original clean, class-balanced distribution, isolating the effect of the training-time shift on stability.

Class-imbalanced training. Following Cao et al. (2019), we reduce the number of training samples per class while keeping the test set unchanged. We sweep two imbalance patterns—long-tailed (exponential decay) Cui et al. (2019) and step imbalance Buda et al. (2018); Azizzadenesheli et al. (2019)—at two ratios (100:1 and 50:1), yielding four imbalanced training sets per benchmark. We initially screened `ResNet-34`, `ResNet-50`, `EfficientNet-b0`, and `ViT-base` on the balanced versions and observed negligible run-to-run variance for all models. Under imbalance, CNN-based models showed lower mean accuracy but only modest variance relative to `ViT-base`, consistent with prior work on long-tailed recognition. We therefore use `ViT-base` for the imbalance experiments, where the seed-induced variance is largest and the stability story is most informative.

Noisy-label training. We corrupt each training label with probability $\rho \in \{0.2, 0.4, 0.6, 0.8\}$ by replacing it with a uniformly-chosen class other than the true one (symmetric label noise). We train `ResNet-18` and `ResNet-32` on CIFAR-10 and CIFAR-100 under each ρ, with the test set left clean. Label noise produces a controlled form of intra-batch gradient conflict—the gradients of correctly- and incorrectly-labeled examples within a mini-batch point in systematically different directions—whose intensity is set by ρ, complementing the imbalance experiments (where conflict is driven by class frequency rather than supervision noise).

We follow Dosovitskiy et al. (2021) recommendation on finetuning `ViT` on CIFAR-10 and CIFAR-100 datasets using SGD with a learning rate of $1E-2$ and a total batch size of 128 detailed setup can be found in Table 8.

E.5 Additional Results and Analysis

E.5.1 Regression Task

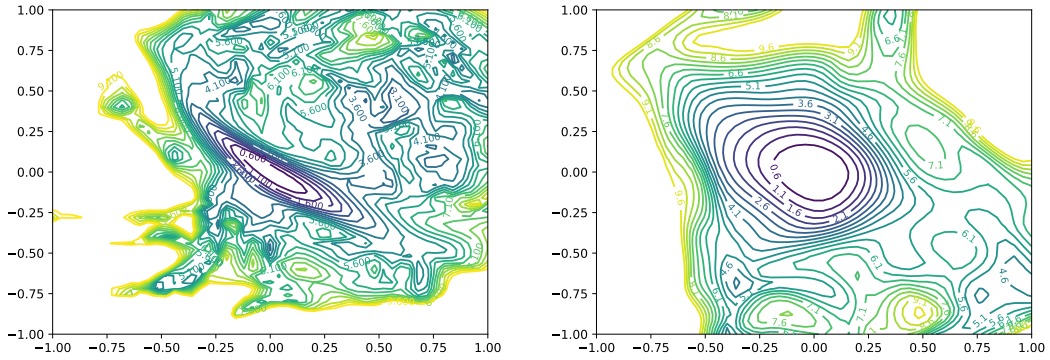
Models: `Roberta-Large`, `Llama-3.2-1B`. **Datasets:** STS-B from GLUE Wang et al. (2018). **Metrics:** Pearson correlation.

Results. Table 2 shows that GRAIN matches the performance of FFT (91.56% versus 91.86% and 90.78% versus 90.12% across the two backbone models) while outperforming all other baselines in semantic similarity regression task. We also evaluate GRAIN on a simple regression task, namely Diabetes Pedregosa et al. (2011), using a two-layer MLP with 32 neurons per layer. GRAIN achieves a lower root mean squared error of $54.04_{\pm 0.96}$, outperforming standard SGD training, which obtains $57.01_{\pm 0.78}$. Across both setups, GRAIN consistently outperforms existing training methods.

E.6 Empirical Evidence on Loss Landscape Sharpness

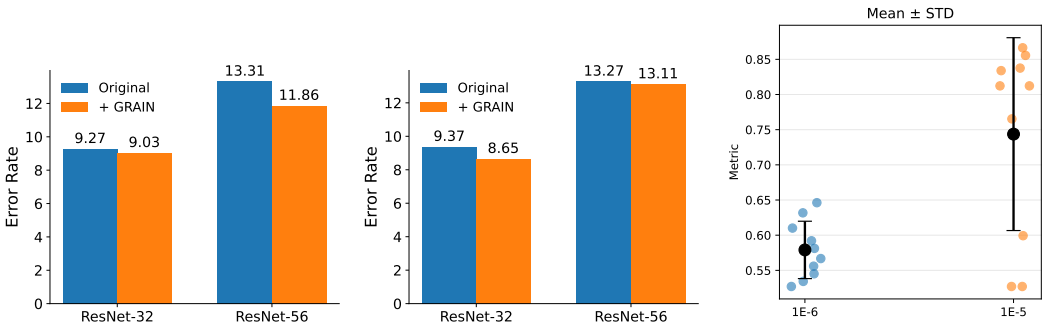
We provide empirical evidence supporting Theorem 4.6. Following the experimental setup in Li et al. (2018), we train `ResNet-32` and `ResNet-56` without residual connections (denoted as `Resnet-32_nr` and `Resnet-56_nr` for short) on CIFAR, both with and without GRAIN, and visualize their loss landscapes. Figure 4 visualizes the loss landscapes with and without GRAIN. It is observable that `Resnet-56_nr` trained without GRAIN exhibit sharper loss landscapes, whereas

GRAIN leads to noticeably smoother optimization surfaces. `Resnet-32_nr` shows similar pattern which is omitted for the sake of brevity. As illustrated in Fig. 4, GRAIN smooths the loss landscape of `ResNet-56_nr` and reduces its error rate from 13.31% to 11.86%. Fig. 5 (a) and (b) present error rate(%) on CIFAR-10 and CIFAR-100 by `ResNet-32_nr` and `ResNet-56_nr`. We observe consistent improvements across tasks and models, with error rate decreases significantly, e.g., 0.72% reduction (9.37% vs 8.64%) on CIFAR-100 by `ResNet-32_nr` and 1.45% reduction (13.31% vs 11.86%) on CIFAR-10 by `ResNet-56_nr`.



(a) **Without** GRAIN. (b) **With** GRAIN.

Figure 4: Loss landscapes on CIFAR-10.



(a) CIFAR-10. (b) CIFAR-100. (c) RTE.

Figure 5: (a) and (b): Test error on CIFAR-{10,100}. (c) Accuracy on RTE with FFT of `RoBERTa-large` across 10 seeds under learning rate 1E−6 and 1E−5. Reducing the learning rate does not remove the failure runs and neither the seed-induced variance, supporting the claim in Section 2 that learning-rate tuning cannot resolve gradient cancellation.

E.6.1 Analysis

Figure 6 presents the performance of individual random seeds for the examined models on both PubMedQA and GSM8K under SGD and GRAIN fine-tuning. Compared to GRAIN, SGD exhibits substantially higher variance, with performance fluctuating significantly across different seeds. This instability is particularly pronounced for `Mistral-7B` and `Mistral-14B` on PubMedQA, where performance ranges from approximately 10% in failed runs to nearly 70% in successful runs.

In contrast, GRAIN greatly reduces this seed sensitivity. For example, the performance of `Mistral-7B` under GRAIN is consistently concentrated around 70%, matching the best performance achieved by SGD while avoiding catastrophic failures. Similar trends are observed across the other models and datasets, indicating that GRAIN not only improves average performance but also substantially enhances training stability and robustness to random initialization.

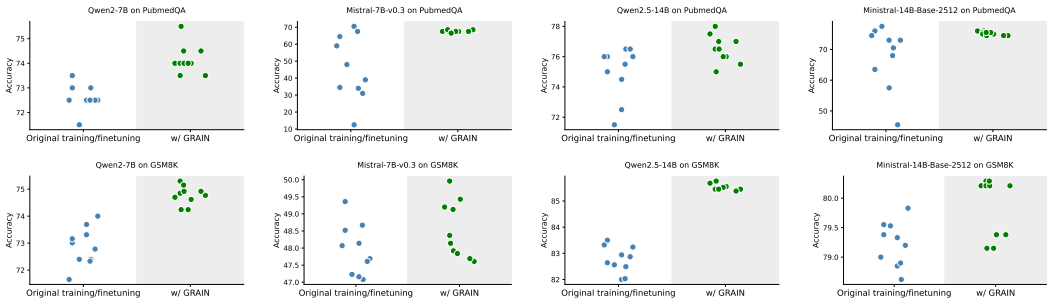


Figure 6: Individual seed performance of `Qwen2-7B`, `Mistral3-7B`, `Qwen2.5-14B` and `Ministral-14B-Base` (left to right) using LoRA finetuning with (green) and with out (blue) using GRAIN on PubmedQA (the first row) and GSM8K (the second row).

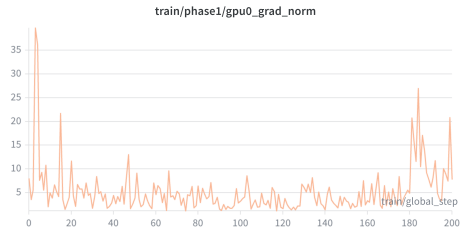
Figures 7 and 8 visualize the gradient dynamics of successful and failed runs at both the local level (within a mini-batch) and the global level (across mini-batches). We observe that the cosine similarity between gradients is consistently higher in successful runs than in failed runs, indicating better gradient alignment throughout training.

At the local level, failed runs exhibit frequent gradient conflicts, leading to severe gradient cancellation. Similar behavior is observed at the global level. Although inter-batch gradient cancellation is less severe than intra-batch cancellation, as evidenced by the substantially lower frequency of negative cosine similarities between g_p and g_c, failed runs still show considerably more gradient conflict than successful runs. In particular, from iterations 20 to 160, failed runs consistently experience inter-batch conflicts, whereas successful runs encounter fewer than five such conflicts, primarily during the early stages of training (iterations 0 to 80), after which the gradients become largely aligned.

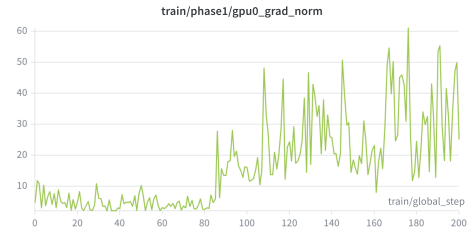
This persistent gradient cancellation in failed runs results in parameter updates with very small magnitudes. Specifically, the norm of the parameter update remains around 5 from iterations 20 to 160. In contrast, successful runs exhibit substantially larger update norms, especially after iteration 80, enabling more effective optimization. Overall, these observations suggest that persistent gradient cancellation at both intra- and inter-batch levels can significantly hinder learning by producing ineffective parameter updates.

	PubmedQA / GSM8K			
	Qwen2-7B	Qwen2.5-14B	Mistral-7B-v0.3	Ministral-14B-Base-2512
	LoRA settings			
Rank	64	64	64	64
Alpha	16	16	16	16
Dropout	0.05	0.05	0.05	0.05
Modules	{ q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj }			
	Others settings			
Learning rate	2E-4			
Batch size	8	4	8	4
Epoch	8 / 3	8 / 3	8 / 3	8 / 3
LR schedule	Cosine	Cosine	Cosine	Cosine
Warmup ratio	0.03	0.03	0.03	0.03
Max new tokens	256/512	256/512	256/512	256/512

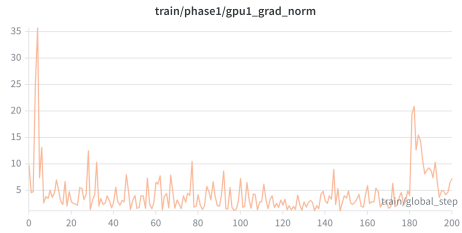
Table 5: Hyperparameter settings for generative tasks.



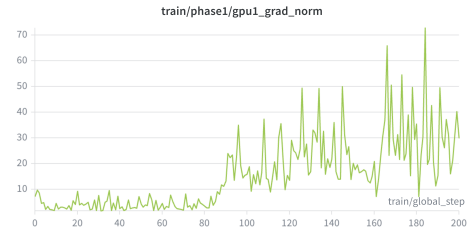
(a) Gradient norm of group $\#1 : g_1$ (Local level - Failed).



(b) Gradient norm of group $\#1 : g_1$ (Local level - Successful).



(c) Gradient norm of group $\#2 : g_2$ (Local level - Failed).



(d) Gradient norm of group $\#2 : g_2$ (Local level - Successful).



(e) Cosine similarity between g_1 and g_1 (Local level - Failed).

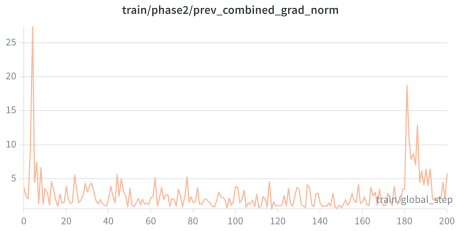


(f) Cosine similarity between g_1 and g_2 (Local level - Successful).

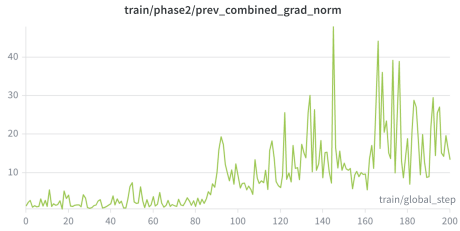
Figure 7: Gradient visualization of a **failed** run (left) and a **successful** run (right) at local level. Result is received from finetuning `Roberta-large` on RTE on 2 GPUs. g_1 is received from GPU$\#1$ and g_2 is received from GPU$\#2$.

	PubmedQA		GSM8K		SuperGLUE		GLUE		CIFAR-10		CIFAR-100		Diabetes	
	m	k	m	k	m	k	m	k	m	k	m	k	m	k
`Qwen2-7B`	4	2	4	2	–	–	–	–	–	–	–	–	–	–
`Qwen2.5-14B`	4	4	4	4	–	–	–	–	–	–	–	–	–	–
`Mistral-7B-v0.3`	4	2	2	1	–	–	–	–	–	–	–	–	–	–
`Ministral-14B-Base-2512`	4	4	4	4	–	–	–	–	–	–	–	–	–	–
`Llama-3.2-1B`	–	–	–	–	2	2	2	2	–	–	–	–	–	–
`RoBERTa-large`	–	–	–	–	2	2	2	2	–	–	–	–	–	–
`ViT-Base`	–	–	–	–	–	–	–	–	2	2	2	2	–	–
`Resnet-18`	–	–	–	–	–	–	–	–	2	1	2	1	–	–
`Resnet-32`	–	–	–	–	–	–	–	–	2	1	2	1	–	–
`Resnet-32_nr`	–	–	–	–	–	–	–	–	2	1	–	–	–	–
`Resnet-56_nr`	–	–	–	–	–	–	–	–	2	1	–	–	–	–
`MLP [32,32]`	–	–	–	–	–	–	–	–	–	–	–	–	2	1

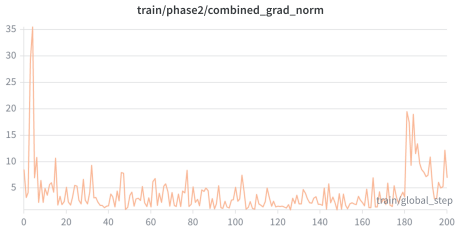
Table 6: Detailed grouping settings of GRAIN for each task. m is the number of intra-batch groups; k is the number of inter-batch consecutive batches.



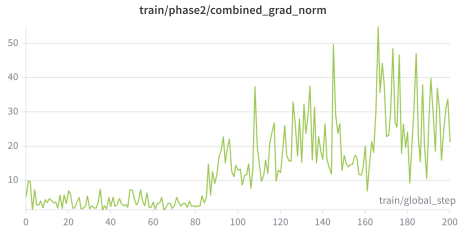
(a) Gradient norm of previous iteration gradient: g_p (Global level - Failed).



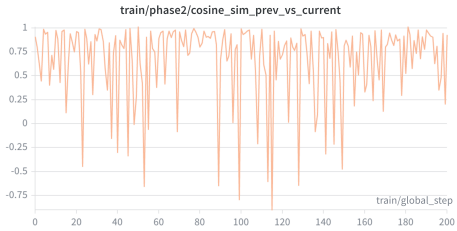
(b) Gradient norm of previous iteration gradient: g_p (Global level - Successful).



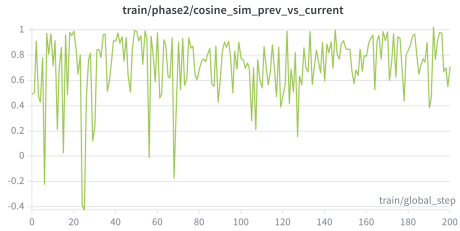
(c) Gradient norm of current iteration gradient: g_c (Global level - Failed).



(d) Gradient norm of current iteration gradient: g_c (Global level - Successful).



(e) Cosine similarity between g_p and g_c (Global level - Failed).



(f) Cosine similarity between g_p and g_c (Global level - Successful).

Figure 8: Gradient visualization of a **failed** run (left) and a **successful** run (right) at global level. Result is received from finetuning `Roberta-large` on RTE on 2 GPUs. g_1 is received from GPU#1 and g_2 is received from GPU#2. Global gradient g_p and g_c are averaged over 2 devices.

		Sequence (Super)GLUE				*Imbalanced CIFAR*	
		RTE	MRPC	BoolQ	STS-B	CIFAR-10	CIFAR-100
Task type		cls.	cls.	cls.	reg.	cls.	cls.
Number of classes		2	2	2	–	10	100
Training samples	Long-tailed (100:1)	–	–	–	–	12,406	10,847
	Long-tailed (50:1)	–	–	–	–	36,223	36,029
	Step (100:1)	–	–	–	–	25,250	25,250
	Step (50:1)	–	–	–	–	37,500	37,500
Training samples (no imbalance)		2,490	3,668	9,427	5,749	–	–
Validation samples		277	408	3,270	1,500	10,000	10,000

Table 7: Dataset statistics. "cls." = classification, "reg." = regression. STS-B values to be filled in. CIFAR train sizes are reported per imbalance pattern (long-tailed and step) at imbalance ratios 100:1 and 50:1 following (Cao et al., 2019); the validation set is held out unchanged.

	RoBERTa-large / Llama-3.2-1B				*ViT-base*	
	RTE	MRPC	BoolQ	STS-B	CIFAR-10	CIFAR-100
Batch size per device	5	5	5	5	64	64
Gradient accumulation	2	2	2	2	2	2
Learning rate	{5E-6, 1E-5, 2E-5, 3E-5, 1E-4, 2E-4, 3E-4}				{1E-1, 1E-2, 1E-3}	
LR scheduler	linear	linear	linear	linear	linear	linear
Max sequence length	512	256	512	256	–	–
Epochs	10	3	10	3	5	5
Number of GPUs	2	2	2	2	2	2

Table 8: Hyperparameter settings for sequence classification/regression (RoBERTa-large and Llama-3.2-1B) and image classification (ViT-base). Values in the LM block apply to both backbones; the ViT block applies to all imbalanced CIFAR variants.