Imbalanced Gradients in RL Post-Training of Multi-Task LLMs

Runzhe Wu[1,3,*], Ankur Samanta[1,4], Ayush Jain[1], Scott Fujimoto[2], Jeongyeol Kwon[1], Ben Kretzu[5], Youliang Yu[1], Kaveh Hassani[2], Boris Vidolov[1], Yonathan Efroni[1,6,*]
[1]Meta AI, [2]Meta Superintelligence Labs, [3]Cornell University,
[4]Columbia University, [5]Technion, [6]Tel Aviv University

Abstract

Multi-task post-training of large language models (LLMs) is typically performed by mixing datasets from different tasks and optimizing them jointly. This approach implicitly assumes that all tasks contribute gradients of similar magnitudes; when this assumption fails, optimization becomes biased toward large-gradient tasks. In this paper, however, we show that this assumption fails in RL post-training: certain tasks produce significantly larger gradients, thus biasing updates toward those tasks. Such gradient imbalance would be justified only if larger gradients implied larger learning gains on the tasks (i.e., larger performance improvements)—but we find this is not true. Large-gradient tasks can achieve similar or even much lower learning gains than small-gradient ones. Further analyses reveal that these gradient imbalances cannot be explained by typical training statistics such as training rewards or advantages, suggesting that they arise from the *inherent* differences between tasks. This cautions against naive dataset mixing and calls for future work on principled gradient-level corrections for LLMs.

1 Introduction

Large language models (LLMs) are increasingly trained to master multiple tasks simultaneously, from language understanding and summarization to code generation and math problem solving. Such multi-task learning enables models to generalize across domains and learn more efficiently than by training separate models per task (Qi et al., 2024; Brief et al., 2024).

A standard approach to multi-task post-training is to merge datasets from different tasks and train on them jointly (Team et al., 2023; Wang et al., 2023a,b; Hu et al., 2024; Zhu et al., 2025). While simple, this strategy effectively mixes gradients across tasks and implicitly requires gradient magnitudes to be similar across tasks. When some tasks generate much larger gradients, the optimization becomes dominated by them. Such imbalance has been noted in the broader field such as vision (Chen et al., 2018) but is less explored in the LLM literature (Section A).

In this paper, we show that such gradient imbalance arises prominently in the RL post-training of multi-task LLMs (Figure 1). The consequence of this imbalance is detrimental: (1) when certain tasks produce much larger gradients, the gradients aggregated from all tasks are dominated by them, which biases updates toward large-gradient tasks and reduces progress on the rest; and (2) from an optimization perspective, large-gradient tasks are effectively trained as if their learning rates were set to be larger than those of small-gradient tasks. Therefore, either large-gradient tasks are optimized too aggressively, or small-gradient tasks are under-optimized.

Crucially, such gradient imbalances would only be reasonable if larger gradients correspond to greater learning gains; i.e., large-gradient tasks should dominate optimization only when they indeed yield larger improvements on their respective objectives (in our case, training rewards). However, we find this hypothesis does not hold: tasks with much larger gradients can exhibit *similar or even lower* learning gains than those with much smaller gradients. To further examine the correlation between gradient magnitude and learning gains, we implement gradient-proportional sampling that allocates more training to large-gradient tasks to see if it brings any advantage—the rationale is that if gradient magnitude truly reflects learning gains, then training more frequently on large-gradient tasks should improve overall average performance by prioritizing tasks with greater gains while sacrificing those with smaller ones (Chen et al., 2025; Wang et al., 2025). However, we found this ap-

[*]Work done at Meta. Correspondence: Runzhe Wu at rw646@cornell.edu.

Findings of the Association for Computational Linguistics: EACL 2026, pages 3137–3150
March 24-29, 2026 ©2026 Association for Computational Linguistics

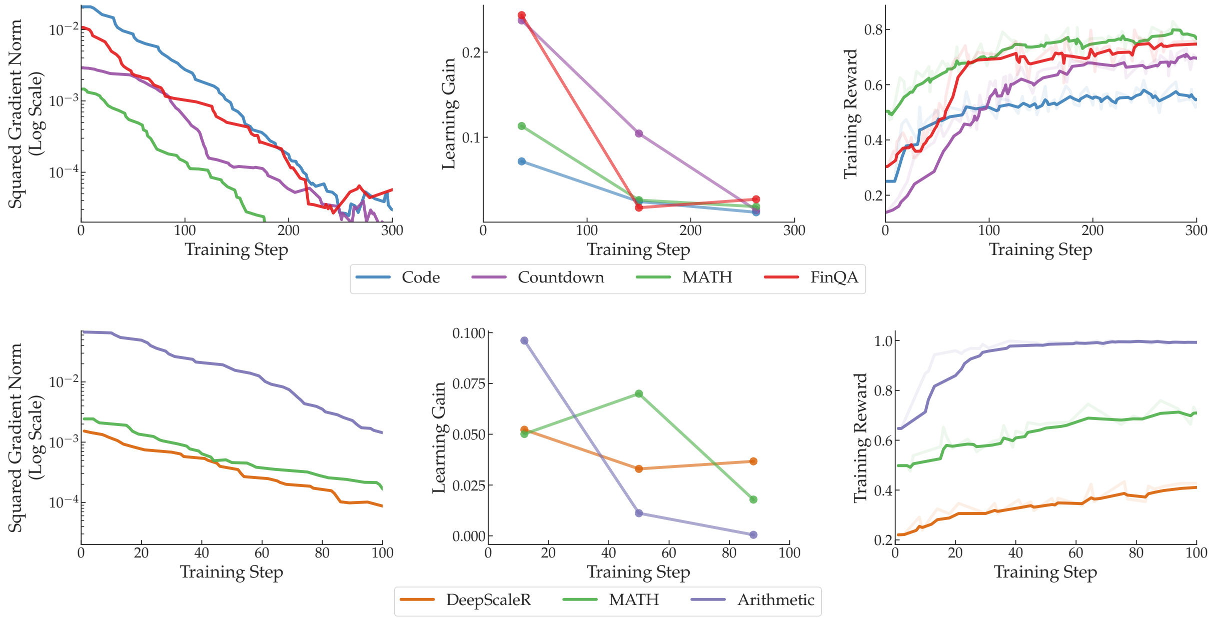

Figure 1: **Gradient imbalance and its misalignment with learning gains.** *Top:* Qwen-7B trained on four heterogeneous tasks. The left panel shows squared gradient norms, with Code dominating throughout training. However, the middle panel reveals that Code achieves the *lowest* learning gains—defined as the increase in training reward (Section 3.1). The right panel reports training rewards for reference. *Bottom:* Qwen-7B trained on three math tasks. Arithmetic exhibits up to 33× larger squared gradients but major learning gains vanish after early training. Together, these results indicate an implicit training bias due to imbalanced gradients which cannot be attributed to learning gains. Plots of other models are provided in Section D.3.

proach shows no advantages and can perform even worse in our experiments, proving that the gradient signal across tasks is not merely uncorrelated with learning gains but misleading when used for training signals. In summary, we find that gradient imbalance is not caused by imbalanced learning gains and can thus harm multi-task learning by biasing updates.

To probe the source of this imbalance, we examined some training statistics as potential explanations, such as training rewards and advantage functions, but we found none could account for it (Section D.1). It suggests that the gradient imbalance does not depend on these training statistics but arises from inherent differences between tasks. This highlights the need for future methods that explicitly correct it in training.

2 Imbalanced Gradients

In this section, we provide the empirical evidence of gradient imbalance in multi-task RL post-training. We study different models: instruction-tuned versions of Qwen2.5-3B and 7B (Qwen et al., 2024) and Llama-3.2-3B (Dubey et al., 2024). We consider RLVR (Guo et al., 2025) and use GRPO (Shao et al., 2024) as the RL algorithm. Additional details of the implementation and the results are reported in Sections C and D. We consider two multi-task settings: (1) four different tasks across multiple domains and (2) three tasks within the math domain but with varying difficulty. They are described below.

Multi-domain tasks. We focus on four tasks: (1) *Code*, involving code generation in the style of programming contests (Liu and Zhang, 2025); (2) *Countdown*, where the model needs to construct a target number from given inputs using basic arithmetic operators (Pan et al., 2025); (3) *MATH*, focused on mathematical problem solving (Hendrycks et al., 2021); and (4) *FinQA*, financial reasoning over tabular data and text (Chen et al., 2021). This setup focuses on examining the gradient behaviors across heterogeneous tasks.

Single-domain (math) tasks. We consider three datasets related to mathematics. *DeepScaleR* (Luo et al., 2025) and *MATH* (Hendrycks et al., 2021) contain challenging math problems (with DeepScaleR being harder overall). *Arithmetic* (Brown et al., 2020) consists of basic arithmetic problems and is easier. This setup allows us to examine gradient behavior within a single domain while varying task difficulty and problem styles.

We conduct experiments for both multi-task settings described above. During training, batches are constructed by sampling from all tasks with equal probability—importantly, we sample with equal probability rather than simply mixing the datasets from all tasks because the latter leads to a natural bias due to different dataset sizes across tasks. To study gradient imbalance, we track the average squared gradient norms of each task throughout training and report them in Figure 1. We focus on *squared* norms rather than unsquared ones because theoretically they are more directly connected to learning gains (Section B), which we will compare against in subsequent sections. For completeness, we also report the unsquared norm in Section D.3 (Figure 5) where we observe the identical trend.

From Figure 1 (left column), we observe that the gradient magnitudes are decreasing as training proceeds for all tasks. However, there is a clear dominance pattern across tasks. For instance, in the multi-domain setting (top left), the gradient of Code has been the largest throughout training—its squared norm is up to $15\times$ larger than MATH; in the single-domain setting (bottom left), Arithmetic has been dominating the others with squared norms up to $33\times$ larger. Such an issue is severe if we consider how gradients are averaged during training. Suppose we have M tasks, each with gradient g_i for $i \in [M]$. A uniform mixture leads to the average gradient $\bar{g} = \frac{1}{M}\sum_{i\in[M]} g_i$. If certain tasks have substantially larger gradients than others (e.g., Arithmetic exhibiting significantly larger gradients than the others), the averaged gradient becomes effectively dominated by those tasks: $\bar{g} \approx \frac{1}{M}\sum_{i\in\mathcal{I}} g_i$ where $\mathcal{I}$ denotes the set of tasks with outstandingly large gradients. As a result, optimization is biased toward these tasks while under-optimizing the rest ($[M] \setminus \mathcal{I}$). Another way to view this effect is through the lens of the learning rate: with a globally fixed learning rate, a task such as Arithmetic (with gradient norms up to $\sqrt{33} \approx 5.7\times$ larger than others) was effectively trained *as if its learning rate were* $5.7\times$ *higher than others*. Hence, to stabilize training, the global learning rate must then be reduced, which in turn causes small-gradient tasks to be under-optimized. This runs counter to the goal of multi-task learning, where all tasks should be treated equally.

3 Are Gradient Imbalances Explained by Learning Gains?

Having established the presence of gradient imbalances, we next ask whether they can be explained by learning gains. The reasoning is simple: in multi-task training, large gradients can be justified only if they faithfully reflect larger room for improvement than the others. Only in this case will the gradient bias be not harmful and even be desirable. This hypothesis underlies several successful approaches in the broader machine learning literature (Settles et al., 2007; Ash et al., 2020; Chen et al., 2025). However, whether it holds in multi-task RL post-training remains to be validated. We examine it in two ways: (1) by explicitly quantifying learning gains as the increase in training reward and comparing it with gradient magnitudes, and (2) by studying a gradient-proportional sampling strategy. Our results show that, while gradient magnitude tends to be correlated with learning gains *within a task*, which is consistent with the prior intuition, it does not hold *across tasks*, which is the central issue highlighted in this paper. We elaborate on the two methods in the following two subsections.

3.1 Quantifying Learning Gains

As a direct approach, we explicitly measure the learning gains as the increase in training reward and compare it with gradient magnitudes to see if they are correlated. Particularly, the learning gain on a given task at training step t is measured as the average change in training reward around t:

$$\text{Gain}(t) := \frac{1}{s}\sum_{i=1}^{s} R_{t+i} - \frac{1}{s}\sum_{i=1}^{s} R_{t-i},$$

where R_k denotes training reward at step k, and s is the window size used to smooth rewards. We evaluate the learning gains at three evenly spaced training steps and plot their corresponding values in Figure 1 (middle column). It reveals a different pattern compared to gradient magnitudes (left column). In particular, in the multi-domain setting (top), Code dominates all others in terms of gradient magnitude, yet its gains are one of the *lowest*. The contrast is clearer in the single-domain setting (bottom): Arithmetic constantly exhibits up to $33\times$ larger squared gradient norms than the others, but it shows the lowest learning gains after half of training. Hence, differences in gradient magnitude

Model	**Grad-Prop**	*Multi-domain*					*Single-domain (math)*			
	Sampling?	**Code**	**Countdown**	**MATH**	**FinQA**	***Avg***	**DSR**	**MATH**	**Arith**	***Avg***
Qwen-3B	✗	35.67	**50.00**	**63.20**	60.60	**52.37**	**34.00**	58.00	98.00	63.33
	✓	35.67	42.50	61.00	**62.67**	50.46	33.40	**61.20**	98.00	**64.20**
Llama-3B	✗	30.34	**42.50**	**49.00**	64.65	**46.62**	**26.20**	45.00	95.60	**55.60**
	✓	**30.76**	40.50	48.20	**65.01**	46.12	23.60	**46.00**	**96.00**	55.20
Qwen-7B	✗	49.16	**59.00**	**71.00**	**68.71**	**61.97**	**41.40**	**68.40**	99.20	**69.67**
	✓	**50.70**	53.00	69.60	68.62	60.48	34.20	64.20	**99.60**	66.00

Table 1: **Uniform vs. gradient-proportional sampling.** Across models and multi-task settings, sampling tasks in proportion to gradient magnitude raises accuracy on some large-gradient tasks (e.g., FinQA, Arithmetic) but can lower overall averages. This confirms that large gradients do not necessarily correspond to greater learning gains, so prioritizing them hurts multi-task learning. (DSR: DeepScaleR, Arith: Arithmetic, Avg: Average)

across tasks cannot be attributed to differences in learning gains.

3.2 Gradient-Proportional Sampling

Another way to examine the hypothesis that the gradient magnitude is positively correlated with larger learning gains is to test algorithms that explicitly exploit large gradient magnitudes. Hence, we study a gradient-proportional sampling strategy: to form the training batches at each iteration, instead of sampling tasks uniformly, we sample from tasks in proportion to their average gradient magnitudes. Specifically, given M tasks, we define the probability of sampling from task $i \in [M]$ as the softmax of the gradient magnitudes: $p_i = \exp(\|g_i\|/\eta)/\sum_j \exp(\|g_j\|/\eta)$ where g_i denotes the average gradient of task i and η is the temperature. To avoid the model getting stuck on high-gradient tasks under low-temperature settings, we truncate p_i at 0.1, ensuring that every task is sampled with at least a 10% probability. The rationale behind gradient-proportional sampling is straightforward: if gradient magnitude truly reflects learning gains, then biasing training toward large-gradient tasks should improve overall average performance by prioritizing tasks with greater gains while sacrificing those with smaller ones.

We report the comparison between gradient-proportional sampling and uniform sampling on both multi-task settings across various models in Table 1. The results show no advantage of this approach. While it primarily raises test accuracy on the large-gradient tasks as expected (e.g., Code, FinQA and Arithmetic, evidenced in Figure 1), it can lead to larger drops on other tasks, thus leading to lower average performance. This again suggests that gradient magnitude is not a reliable indicator of learning gains: allocating more training to large-gradient tasks does not consistently improve overall outcomes and may even exacerbate gradient bias. Importantly, our intention is not to claim that gradient-proportional sampling is consistently worse than uniform sampling; if that were the case, one might conclude gradient magnitude is inversely correlated with learning gains, which is counterintuitive and unlikely. Rather, we aim to show that gradient magnitude cannot faithfully represent learning gains. Detailed training curves and ablations of gradient-proportional sampling are provided in Sections D.2 and D.3 for reference.

Summary. Through explicit measurement (Section 3.1) and implicit testing (Section 3.2), we show gradient magnitude does not correlate with learning gains across tasks. We also provide a theoretical analysis in Section B explaining why learning gains do not necessarily correlate with gradient magnitude across tasks from the perspective of convex analysis.

4 Conclusion

We presented the first systematic study of gradient imbalance in the RL post-training phase of multi-task LLMs. Results show a surprising finding: the gradient imbalance is significant and cannot be explained by learning gains. Further, we study whether the gradient imbalances can be explained by certain training statistics or metrics (e.g., advantages, training reward, token length, etc.) but we find no clear correlation. Hence, the observed gradient imbalance likely stems from inherent differences between tasks.

Looking ahead, our findings call for approaches that explicitly perform gradient-level manipulation to address the imbalances, possibly building on ideas from the broader optimization literature (Sener and Koltun, 2018; Yu et al., 2020; Chen et al., 2020; Liu et al., 2021a,b, 2023; Efroni et al., 2025; Kretzu et al., 2025). Beyond that, another promising direction lies in reconsidering the optimization geometry. Methods such as mirror descent-based RL approaches (Kakade, 2001; Gao et al., 2024) may help alleviate this issue by transforming gradients into a certain representation that is more comparable across tasks. Finally, investigating whether the same issue occurs in other training paradigms such as pre-training and supervised finetuning remains important for future work.

Limitations

This paper focuses primarily on the RL post-training of multi-task LLMs. It remains unclear whether gradient imbalance also persists in other phases, such as pre-training and supervised fine-tuning. Although such imbalances appear to be from the inherent differences between tasks, we cannot yet explain why it arises, despite studying a broad range of training statistics. Moreover, we only report these observations; how to design new approaches to mitigate them remains unclear and requires further research. Finally, we do not anticipate any immediate ethical or societal risk from this work, though broader concerns may emerge from the use of LLMs more generally.

References

Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. 2020. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations*.

Meni Brief, Oded Ovadia, Gil Shenderovitz, Noga Ben Yoash, Rachel Lemberg, and Eitam Sheetrit. 2024. Mixing it up: The cocktail effect of multi-task fine-tuning on llm performance–a case study in finance. *arXiv preprint arXiv:2410.01109*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners.

Xiaoyin Chen, Jiarui Lu, Minsu Kim, Dinghuai Zhang, Jian Tang, Alexandre Piché, Nicolas Gontier, Yoshua Bengio, and Ehsan Kamalloo. 2025. Self-evolving curriculum for llm reasoning. *arXiv preprint arXiv:2505.14970*.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International conference on machine learning*, pages 794–803. PMLR.

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

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan R Routledge, and 1 others. 2021. Finqa: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3697–3711.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Yonathan Efroni, Ben Kretzu, Daniel Jiang, Jalaj Bhandari, Zheqing Zhu, and Karen Ullrich. 2025. Aligned multi objective optimization. In *Forty-second International Conference on Machine Learning*.

Ahmed Elbakary, Chaouki Ben Issaid, Tamer ElBatt, Karim Seddik, and Mehdi Bennis. 2025. Mira: A method of federated multi-task learning for large language models. *IEEE Networking Letters*.

Wenfeng Feng, Chuzhan Hao, Yuewei Zhang, Yu Han, and Hao Wang. 2024. Mixture-of-loras: An efficient multitask tuning method for large language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 11371–11380.

Zhaolin Gao, Jonathan Chang, Wenhao Zhan, Owen Oertell, Gokul Swamy, Kianté Brantley, Thorsten Joachims, Drew Bagnell, Jason D Lee, and Wen Sun. 2024. Rebel: Reinforcement learning via regressing relative rewards. *Advances in Neural Information Processing Systems*, 37:52354–52400.

Zhaolin Gao, Joongwon Kim, Wen Sun, Thorsten Joachims, Sid Wang, Richard Yuanzhe Pang, and Liang Tan. 2025. Prompt curriculum learning for efficient llm post-training. *arXiv preprint arXiv:2510.01135*.

Zi Gong, Hang Yu, Cong Liao, Bingchang Liu, Chaoyu Chen, and Jianguo Li. 2024. Coba: Convergence balancer for multitask finetuning of large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8063–8077.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Elad Hazan and 1 others. 2016. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325.

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

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, and 1 others. 2024. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*.

Sham M Kakade. 2001. A natural policy gradient. *Advances in neural information processing systems*, 14.

Ben Kretzu, Karen Ullrich, and Yonathan Efroni. 2025. Simple optimizers for convex aligned multi-objective optimization. *arXiv preprint arXiv:2509.05811*.

Yiqing Liang, Jielin Qiu, Wenhao Ding, Zuxin Liu, James Tompkin, Mengdi Xu, Mengzhou Xia, Zhengzhong Tu, Laixi Shi, and Jiacheng Zhu. 2025. Modomodo: Multi-domain data mixtures for multimodal llm reinforcement learning. *arXiv preprint arXiv:2505.24871*.

Bo Liu, Yihao Feng, Peter Stone, and Qiang Liu. 2023. Famo: Fast adaptive multitask optimization. *Advances in Neural Information Processing Systems*, 36:57226–57243.

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

Jiawei Liu and Lingming Zhang. 2025. Code-r1: Reproducing r1 for code with reliable rewards.

Liyang Liu, Yi Li, Zhanghui Kuang, Jing-Hao Xue, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. 2021b. Towards impartial multi-task learning. In *International Conference on Learning Representations*.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica. 2025. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. Notion Blog.

Yuren Mao, Zekai Wang, Weiwei Liu, Xuemin Lin, and Pengtao Xie. 2022. MetaWeighting: Learning to weight tasks in multi-task learning. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3436–3448, Dublin, Ireland. Association for Computational Linguistics.

Jiayi Pan, Junjie Zhang, Xingyao Wang, Lifan Yuan, Hao Peng, and Alane Suhr. 2025. Tinyzero. https://github.com/Jiayi-Pan/TinyZero. Accessed: 2025-01-24.

Shubham Parashar, Shurui Gui, Xiner Li, Hongyi Ling, Sushil Vemuri, Blake Olson, Eric Li, Yu Zhang, James Caverlee, Dileep Kalathil, and 1 others. 2025. Curriculum reinforcement learning from easy to hard tasks improves llm reasoning. *arXiv preprint arXiv:2506.06632*.

Boris Teodorovich Polyak. 1963. Gradient methods for minimizing functionals. *Zhurnal vychislitel'noi matematiki i matematicheskoi fiziki*, 3(4):643–653.

Zhen Qi, Jiajing Chen, Shuo Wang, Bingying Liu, Hongye Zheng, and Chihang Wang. 2024. Optimizing multi-task learning for enhanced performance in large language models. In *2024 4th International Conference on Electronic Information Engineering and Computer Communication (EIECC)*, pages 1179–1183. IEEE.

A Yang Qwen, Baosong Yang, B Zhang, B Hui, B Zheng, B Yu, Chengpeng Li, D Liu, F Huang, H Wei, and 1 others. 2024. Qwen2. 5 technical report. *arXiv preprint*.

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

Burr Settles, Mark Craven, and Soumya Ray. 2007. Multiple-instance active learning. *Advances in neural information processing systems*, 20.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2024. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*.

Taiwei Shi, Yiyang Wu, Linxin Song, Tianyi Zhou, and Jieyu Zhao. 2025. Efficient reinforcement finetuning via adaptive curriculum learning. *arXiv preprint arXiv:2504.05520*.

Richard S Sutton, Andrew G Barto, and 1 others. 1998. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.

MosaicML NLP Team and 1 others. 2023. Introducing mpt-7b: A new standard for open-source, commercially usable llms. *Retrieved December*, 6:2023.

Kaiwen Wang, Rahul Kidambi, Ryan Sullivan, Alekh Agarwal, Christoph Dann, Andrea Michi, Marco Gelmi, Yunxuan Li, Raghav Gupta, Kumar Avinava Dubey, Alexandre Rame, Johan Ferret, Geoffrey

Cideron, Le Hou, Hongkun Yu, Amr Ahmed, Aranyak Mehta, Leonard Hussenot, Olivier Bachem, and Edouard Leurent. 2024. Conditional language policy: A general framework for steerable multi-objective finetuning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 2153–2186, Miami, Florida, USA. Association for Computational Linguistics.

Rongsheng Wang, Haoming Chen, Ruizhe Zhou, Yaofei Duan, Kunyan Cai, Han Ma, Jiaxi Cui, Jian Li, Patrick Cheong-Iao Pang, Yapeng Wang, and 1 others. 2023a. Aurora: Activating chinese chat capability for mixtral-8x7b sparse mixture-of-experts through instruction-tuning. *arXiv preprint arXiv:2312.14557*.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Chandu, David Wadden, Kelsey MacMillan, Noah A Smith, Iz Beltagy, and 1 others. 2023b. How far can camels go? exploring the state of instruction tuning on open resources. *Advances in Neural Information Processing Systems*, 36:74764–74786.

Zhenting Wang, Guofeng Cui, Yu-Jhe Li, Kun Wan, and Wentian Zhao. 2025. Dump: Automated distribution-level curriculum learning for rl-based llm post-training. *arXiv preprint arXiv:2504.09710*.

Rui Yang, Xiaoman Pan, Feng Luo, Shuang Qiu, Han Zhong, Dong Yu, and Jianshu Chen. 2024. Rewards-in-context: Multi-objective alignment of foundation models with dynamic preference adjustment. In *International Conference on Machine Learning*, pages 56276–56297. PMLR.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, and 1 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

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

Ruiqi Zhang, Daman Arora, Song Mei, and Andrea Zanette. 2025. Speed-rl: Faster training of reasoning models via online curriculum learning. *arXiv preprint arXiv:2506.09016*.

Tong Zhu, Daize Dong, Xiaoye Qu, Jiacheng Ruan, Wenliang Chen, and Yu Cheng. 2025. Dynamic data mixing maximizes instruction tuning for mixture-of-experts. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1663–1677.

Stanisław Łojasiewicz. 1963. Une propriété topologique des sous-ensembles analytiques réels. In *Les Équations aux Dérivées Partielles*, Colloques Internationaux du CNRS, pages 87–89. Éditions du CNRS.

A Prior Works on Multi-Task LLMs

Prior works on multi-task LLMs have approached the problem from task weighting and data mixture design (Mao et al., 2022; Gong et al., 2024; Wang et al., 2024; Yang et al., 2024; Feng et al., 2024; Elbakary et al., 2025; Liang et al., 2025). Typically, they adaptively adjust task contributions or update directions to balance convergence across tasks.

Curriculum-based methods have also emerged in the context of LLMs, which schedule tasks adaptively based on a certain notion of learning progress or difficulty (Chen et al., 2025; Shi et al., 2025; Parashar et al., 2025). For instance, it may suggest to train on progressively harder reasoning instances. While these approaches are usually proposed under a single task and manipulate at a sample-level, their natural extension can be applied to the multi-task settings.

In contrast, our study focuses on a fundamental issue: the imbalance of gradient magnitudes across tasks during RL post-training. While similar concerns have been observed in the broader field such as vision (Chen et al., 2018), this issue has received little attention in the context of LLMs.

B Gradient Magnitude and Learning Gains Through the Lens of Convex Analysis

In this section, we study the relationship between gradient magnitude and the learning gain through the lens of convex analysis. Specifically, we will explain why gradient magnitude (squared gradient norm) is expected to be proportional to learning gains *within a task* but not *across tasks* from a theoretical perspective.

Let's think about a single task and thus a single objective optimization problem. Denote the objective function as f. To facilitate the discussion, we consider two common properties of convex functions: smoothness and Polyak-Łojasiewicz condition (a relaxation of strong convexity).

Definition 1 (Smoothness) *A function* $f : \mathbb{R}^n \to \mathbb{R}$ *is said to be* β-smooth *if for all* $x, y \in \mathbb{R}^n$,

$$f(y) \leq f(x) + \nabla f(x)^\top (y - x) + \frac{\beta}{2}\|y - x\|_2^2.$$

It is known that smoothness relates the gradient magnitude to the suboptimality gap (Hazan et al., 2016):

$$f(x) - \min_{x^\star} f(x^\star) \geq \frac{1}{2\beta}\|\nabla f(x)\|_2^2. \qquad (1)$$

Another commonly used property is the Polyak-Łojasiewicz condition:

Definition 2 (Polyak-Łojasiewicz) *(Łojasiewicz, 1963; Polyak, 1963) A function $f : \mathbb{R}^n \to \mathbb{R}$ is said to be μ-*Polyak–Łojasiewicz *(μ-*PL*) if for all $x \in \mathbb{R}^n$, we have*

$$\frac{1}{2}\|\nabla f(x)\|_2^2 \geq \mu \cdot \big(f(x) - \min_{x^\star} f(x^\star)\big). \qquad (2)$$

PL condition generalizes strong convexity (Hazan et al., 2016). By definition, it relates the gradient magnitude to the suboptimality gap in a reversed direction. By (1) and (2), one can establish the following ratio bound:

$$2\mu \leq \frac{\|\nabla f(x)\|_2^2}{f(x) - \min_{x^\star} f(x^\star)} \leq 2\beta$$

This inequality suggests that the squared gradient norm should be proportional to the suboptimality gap. However, estimating the gap in practice is difficult since it requires access to the global minimizer $\min_{x^\star} f(x^\star)$. A useful observation is that performance improvements (i.e., learning gain) during training often follow an exponential decay: once the suboptimality gap becomes small, further gains diminish. Thus, the learning gain serves as a practical proxy for the suboptimality gap, implying that *the magnitude of squared gradient norm should be proportional to learning gain*. This is the fundamentals of certain existing works that leverage gradient magnitude to estimate learning gains.

However, from a theoretical perspective, this proportionality holds only within a single task. When comparing across tasks, the relationship breaks down because each task may have different smoothness β and PL constant μ. This heterogeneity explains the mismatch between gradient magnitude and learning gain across tasks observed in Figure 1.

C Experimental Details

This section details the experiment setup. All implementations are built upon the verl framework (Sheng et al., 2024).

Prompt Template. We use the same prompt template for all tasks and all models. Given a `question`, the prompt template is as follows:

```
<|system|>
You are a helpful assistant. The
    user will ask you a question
    and you as the assistant solve
     it. The assistant first
    thinks how to solve the task
    step by step and then provides
     the user with the final
    answer. The final answer must
    be enclosed within <answer
    >...</answer> tag, which
    should appear at the end of
    your reply.
<|user|>
{question}
<|Assistant|>
```

Reward Details. We use a rule-based reward function. A reward of 0.0 is assigned to a response if the format is wrong (e.g., answer tag `<answer>...</answer>` is not found). Otherwise, if the format is correct but answer is wrong, we assign a reward of 0.1. If both are correct, we assign a reward of 1.0.

Model and Dataset Details. We have listed the models and datasets we used in Section 2. For each task/dataset, the data is split into training and test sets (with MATH500 serving as the test set for MATH). All models and data we used are publicly accessible and authorized for our intended use, and the data did not contain personal identifying info or offensive content. Test accuracy is simply measured by pass@1.

Plotting details. In Figure 1, we smooth the squared gradient norm (left plots) via exponential moving average with coefficient 0.9 for better illustration; for the learning gains (middle plots), we smooth rewards over $s = 75$ steps for multi-domain setting and $s = 25$ for single-domain setting; the reward plots on the right are smoothed with coefficient 0.7 with the raw data plotted as the lighter curves. Same also apply to Figure 6 and Figure 4. Figure 2 is plotted by uniformly picking points throughout training and compute the absolute advantage and gradient norm at those points (without smoothing). Figure 3 is plotted with smoothing factor 0.5 with the raw data plotted as the lighter curves. Figure 7 is plotted without smoothing.

Hyperparameters. Hyperparameters are listed in Table 2 and are kept consistent across all training runs. Training was performed via GRPO on A100 GPUs. For both multi-domain and single-domain settings, we train the model until observed conver-

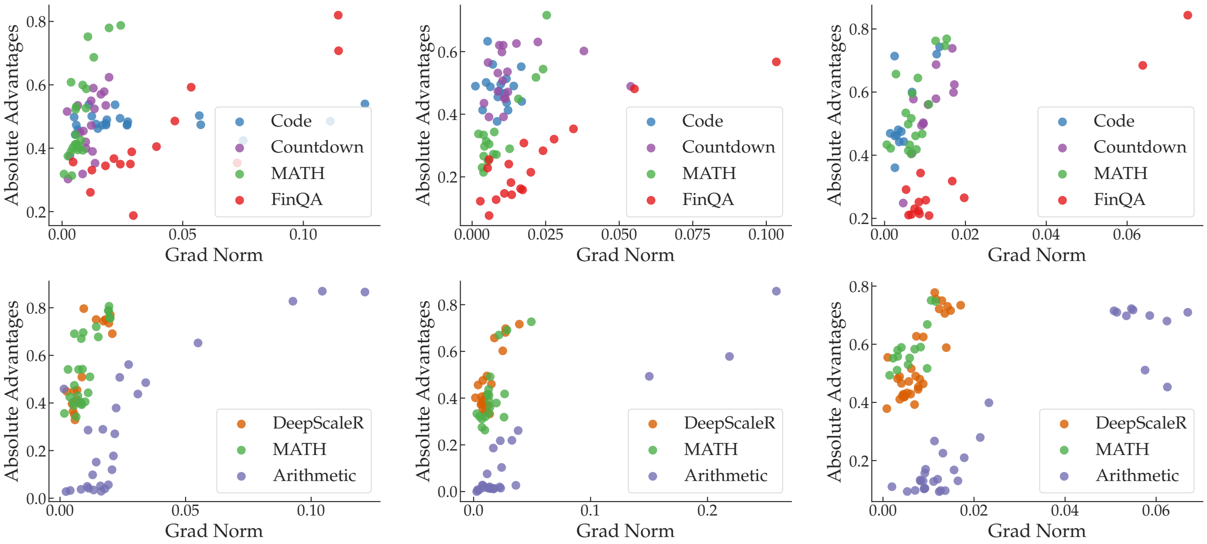

Figure 2: **Gradient norms versus absolute advantage.** Scatter plots across models (from left to right: Qwen-3B, Qwen-7B, Llama-3B) and domains (from top to down: multi-domain and math-domain). Within each task, points tend to cluster and vary consistently. Within a task, larger gradients often align with larger absolute advantages, although unnecessarily linearly. Across tasks, however, this structure disappears: tasks with the largest gradients (e.g., FinQA) do not yield higher absolute advantages than other tasks. Thus, absolute advantage may approximate gradient magnitude *within* a task but fails *across* tasks.

gence (300 steps for the former and 100 steps for the latter).

Parameter	**Value**
batch size	128
rollouts per prompt	16
max prompt length	2048
max response length	4096
learning rate	5e-7
KL coefficient	1e-3
entropy coefficient	1e-3
grad clip	1.0
clip ratio	0.2

Table 2: Training hyperparameters.

C.1 Gradient Estimation

In this section, we explain how we estimate the gradient magnitude in the experiments (e.g., in Figure 1). To obtain a faithful estimator, we cannot simply take the norm of a batch gradient returned by back-propagation. To see why, let g denote the ground truth gradient we want to estimate:

$$g \;=\; \nabla_\theta J(\theta) \;=\; \mathbb{E}[\,X\,],$$

$$\text{where} \quad X \;:=\; \nabla_\theta \log \pi_\theta(a \mid s)\, A^\pi(s,a).$$

Given a batch of size B, the naive plug-in estimator is

$$\hat g \;=\; \frac{1}{B}\sum_{i=1}^{B} X_i, \qquad X_1,\dots,X_B \overset{\text{i.i.d.}}{\sim} X.$$

While $\hat g$ is an unbiased estimator of g, its *squared norm* is a biased estimator of $\|g\|^2$. To see this, using the identity:

$$\mathbb{E}\big[\|\hat g\|^2\big] \;=\; \big\|\mathbb{E}[\hat g]\big\|^2 \;+\; \mathrm{Tr}\big(\mathrm{Cov}(\hat g)\big),$$

we obtain

$$\begin{aligned}\mathbb{E}\big[\|\hat g\|^2\big] \;&=\; \|g\|^2 \;+\; \mathrm{Tr}\big(\mathrm{Cov}(\hat g)\big)\\ &=\; \|g\|^2 \;+\; \frac{1}{B}\,\mathrm{Tr}\big(\Sigma\big),\end{aligned}$$

where $\Sigma := \mathrm{Cov}(X)$. Thus, the naive estimator $\|\hat g\|^2$ is *off from the target* $\|g\|^2$ by the variance term $\mathrm{Tr}(\mathrm{Cov}(\hat g)) = \frac{1}{B}\mathrm{Tr}(\Sigma)$. This term is large considering the huge number of parameters in LLMs.

To remove this bias, we form two independent estimates by splitting the batch:

$$\hat g_1 \;=\; \frac{2}{B}\sum_{i=1}^{B/2} X_i, \qquad \hat g_2 \;=\; \frac{2}{B}\sum_{i=B/2+1}^{B} X_i,$$

and use the cross product

$$\widehat{\|g\|^2} \;=\; \langle \hat g_1, \hat g_2\rangle.$$

Since $\hat{g}_1$ and $\hat{g}_2$ are independent and $\mathbb{E}[\hat{g}_1] = \mathbb{E}[\hat{g}_2] = g$, we have the unbiasedness:

$$\mathbb{E}[\,\langle \hat{g}_1, \hat{g}_2\rangle\,] = \big\langle \mathbb{E}[\hat{g}_1],\, \mathbb{E}[\hat{g}_2]\big\rangle = \langle g, g\rangle = \|g\|^2.$$

Hence, such cross-product estimator provides a faithful (and importantly, unbiased) estimate of the squared gradient norm. In practice, we track the squared gradient norm via the above estimator per step and keep an exponential moving average (with coefficient 0.95) of the tracked values as the final estimate. Furthermore, when we want to estimate the (unsquared) gradient norm, we simply take the square root of the cross-product estimator above, truncated at zero:

$$\hat{\|g\|} \;=\; \sqrt{\max\left(\widehat{\|g\|^2}, 0\right)}.$$

Here we truncate at zero to avoid negative values inside the square root since the cross-product estimator is not guaranteed to be non-negative. To reduce the overall computational cost of the cross product, we only compute the gradient of the last attention block as a proxy for the entire model. An ablation study on this choice is provided in Section D.2.

D Additional Experiment Results

D.1 Are Gradient Imbalances Explained by Other Training Statistics?

Here we study whether gradient magnitude correlates with several key training statistics and metrics, including advantage function, training reward, and token length. We note that, even if such correlations exist, gradient imbalance would remain problematic because it cannot reflect learning gains, as we discussed in Section 3.

Advantage Function. Prior work has speculated that the absolute advantage function can serve as an effective proxy for gradient magnitude of language models, mainly focusing on single-task training (Chen et al., 2025; Wang et al., 2025; Gao et al., 2025). To see the intuition, we recall the policy-gradient theorem. For policy-gradient-based RL, the derivative of the model parameters is derived as (Sutton et al., 1998)

$$\nabla_\theta J(\theta) = \mathbb{E}[\nabla_\theta \log \pi_\theta(a \mid s) A^\pi(s, a)]$$

where $J(\theta)$ denotes the average reward of the policy π_θ, and $A^\pi(s, a)$ is the advantage function. By Jensen's inequality, we have

$$\|\nabla_\theta J(\theta)\|_2 \le \mathbb{E}\big[|A^\pi(s, a)|\|\nabla_\theta \log \pi_\theta(a \mid s)\|_2\big]$$

Hence, $|A^\pi|$ seems to be an effective proxy for gradient magnitude, assuming the inequality is nearly tight and $\nabla_\theta \log \pi_\theta$ does not vary too much. While this hypothesis has led to successful approaches in single-task training, it remains to be seen whether it holds in multi-task training. Thus, we put it to the test and report results in Figure 2, where in the scatter plot, we track and plot the absolute advantage and gradient norm periodically throughout training. The computation of advantage is following standard GRPO, which is estimated via empirical mean and then normalized. Interestingly, we find a peculiar pattern: the absolute advantage is a reasonable proxy for gradient magnitude *within a task*, but it fails to generalize *across tasks*. In particular, within each task, the absolute advantage grows roughly proportionally with the gradient magnitude. However, when comparing across tasks, the same level of absolute advantage corresponds to very different gradient norms: for example, MATH tends to yield much smaller gradients than FinQA. Hence, we conclude that the relationship is valid only within a single task.

Training Reward (Accuracy). It is also tempting to associate gradient magnitude with task difficulty. For example, we may speculate that easy tasks (high average reward/accuracy) yield clearer, larger gradients while hard tasks (low average reward/accuracy) produce noisier, smaller gradients on average. However, this intuition may not hold in the extreme: when accuracy is near 0% or 100%, the advantage A^π is close to zero; by the policy-gradient theorem the expected gradient also approaches zero (Yu et al., 2025; Zhang et al., 2025). Thus, gradient magnitude need not increase monotonically with "easiness" or "hardness". Qualitatively, in Figure 1 (right columns), while within each task, the trend shows gradient norms decreasing as training reward (accuracy) increases, there is no consistent cross-task pattern, suggesting that there is unlikely to be a simple relationship between gradient magnitude and task difficulty.

Prompt/Response Length. Since different tasks have varying prompt and response lengths, we also investigate whether gradient magnitude is affected by this factor. In principle, it should not be, because the policy-gradient loss is averaged over tokens and is therefore expected to be invariant to

	Task	**Avg. Prompt Length**
Multi-domain	Code	553.2
	Countdown	164.5
	MATH	168.6
	FinQA	1161.9
Single-domain	DeepScaleR	173.1
	MATH	168.6
	Arithmetic	105.4

Table 3: Average prompt lengths of all tasks.

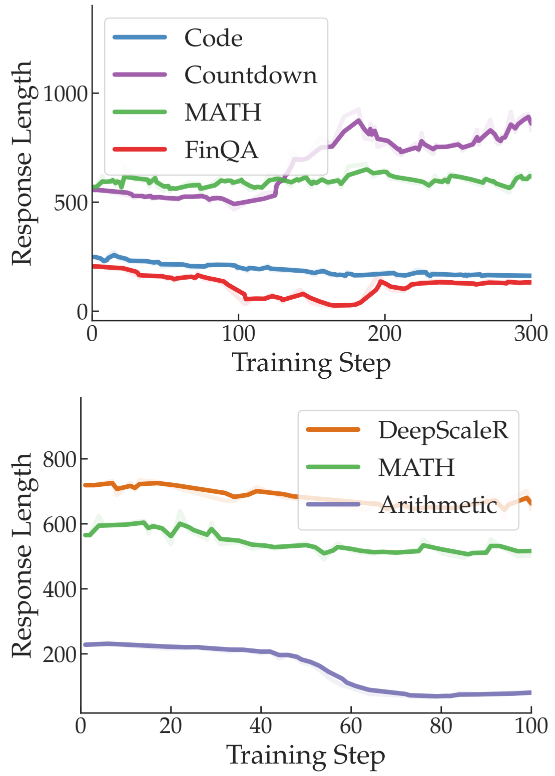

Figure 3: Response length for Qwen-3B model.

sequence length. Nonetheless, we examine the relationship empirically. The average prompt lengths for all tasks are reported in Table 3. We find no meaningful correlation between prompt length and gradient magnitude. FinQA often has long prompts (due to embedded table data) and tends to exhibit large gradients, yet this pattern does not hold universally: in the single-domain setting, Arithmetic has the largest gradient magnitude despite having the shortest average prompt. Similarly, the curves of response length shown in Figure 3 reveal no meaningful connection either.

Summary In summary, we found that the gradient imbalance cannot be easily explained by any of the above training statistics. Hence, the imbalance is likely from the inherent differences between tasks.

D.2 Ablation Study

Gradient-Proportional Sampling. We perform a sweep of temperature for gradient-proportional sampling on Qwen-3B and Llama-3B for multi-domain tasks and report the results in Table 4. For $\eta \geq 0.1$, we find the resulting weights are nearly identical to uniform sampling, so meaningful weight differences appear only when $\eta < 0.1$, which is why we mainly focus on the range below $\eta = 0.1$. However, as the temperature decreases, we consistently observe degraded average performance, indicating that over-weighting large-gradient tasks is detrimental. We fix $\eta = 0.01$ to report results in Table 1 in the main text across all models and settings.

Gradients of Different Attention Blocks. To reduce computational cost, we approximate the full-model gradient by computing it only on the last attention block, as described in Section C.1. To validate this choice, we examine the Qwen-3B model by computing gradients across other layers (first, middle, last) and reporting them in Figure 4. The results show that gradients across layers are of comparable scale and the dominance pattern remains consistent across layers: Code and FinQA have the largest gradients, while MATH has the smallest. Notably, earlier blocks (first and middle blocks) exhibit higher variance. These confirm that the last attention block serves as a reliable proxy.

D.3 Supplementary Plots

In this section, we supplement additional plots of other models and statistics for completeness. The observations are consistent with the main text.

Model	Grad-Prop Sampling?	Temperature	*Multi-domain*				
			Code	**Countdown**	**MATH**	**FinQA**	***Avg***
Qwen-3B	✗	-	35.67	50.00	63.20	60.60	**52.37**
	✓	0.1	35.53	48.50	60.00	61.59	51.41
	✓	0.01	35.67	42.50	61.00	62.67	50.46
	✓	0.001	38.06	37.50	62.60	62.85	50.25
Llama-3B	✗	-	30.34	42.50	49.00	64.65	**46.62**
	✓	0.1	28.51	40.00	44.20	62.04	43.69
	✓	0.01	30.76	40.50	48.20	65.01	46.12
	✓	0.001	29.92	39.50	46.80	64.47	45.17
Qwen-7B	✗	-	49.16	59.00	71.00	68.71	**61.97**
	✓	0.1	50.98	56.50	70.20	68.44	61.53
	✓	0.01	50.70	53.00	69.60	68.62	60.48
	✓	0.001	52.25	52.00	71.00	69.52	61.19

Table 4: **Sweep of gradient-proportional sampling temperature across models.** For temperature $\eta \geq 0.1$, we find the resulting weights are nearly identical to uniform sampling, so we mainly focus on the range below 0.1. As the temperature decreases, we consistently observe degraded average performance, indicating that over-weighting large-gradient tasks is detrimental. This further demonstrates the pitfalls of naively treating gradient magnitude as a proxy for learning gains.

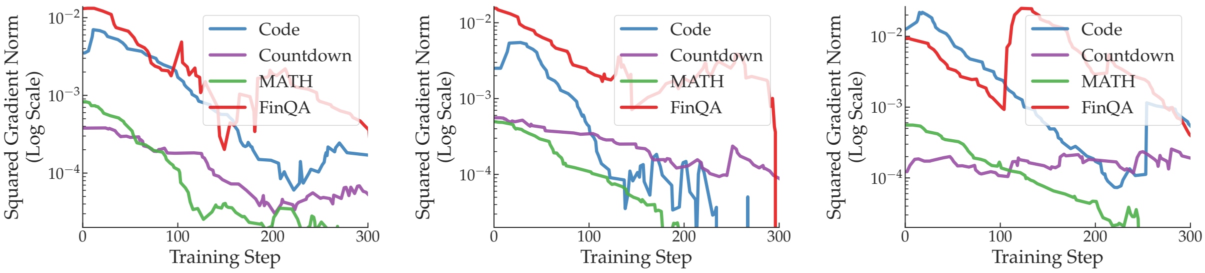

Figure 4: **Gradient norm of different attention blocks for Qwen-3B model on multi-domain tasks.** There are a total of 36 attention blocks in the model. From left to right are the gradients of the last (36th) attention block, middle (18th) attention block, and the first attention block.

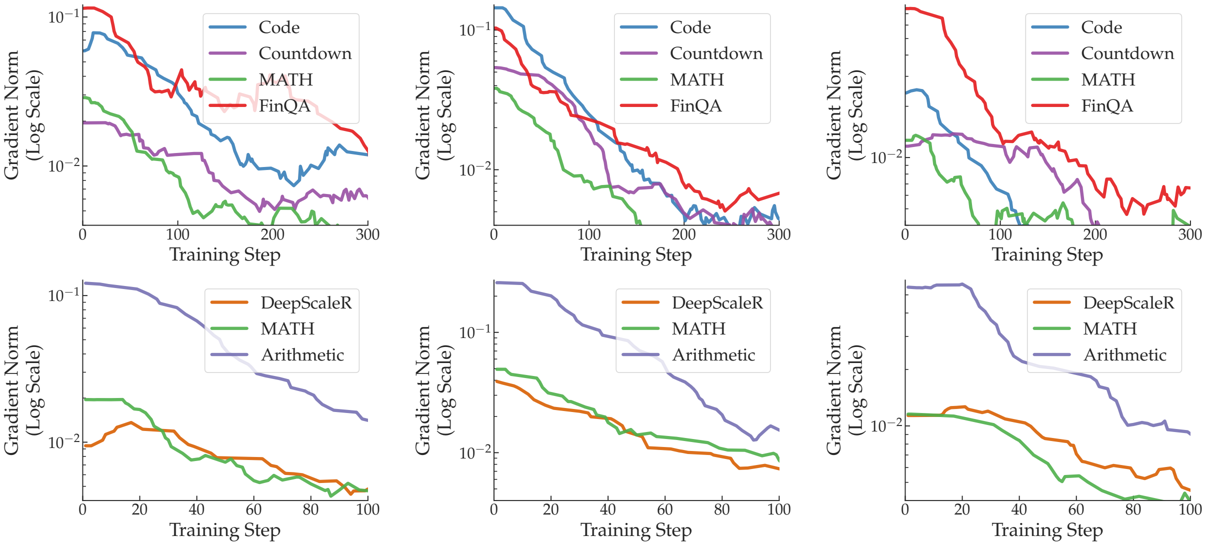

Figure 5: **(Unsquared) Gradient norm across models.** From left to right are Qwen-3B, Qwen-7B, and Llama-3B. Top row presents multi-domain tasks, and bottom row presents single-domain tasks. The pattern is identical to that of the squared norm in Figure 1 and Figure 6.

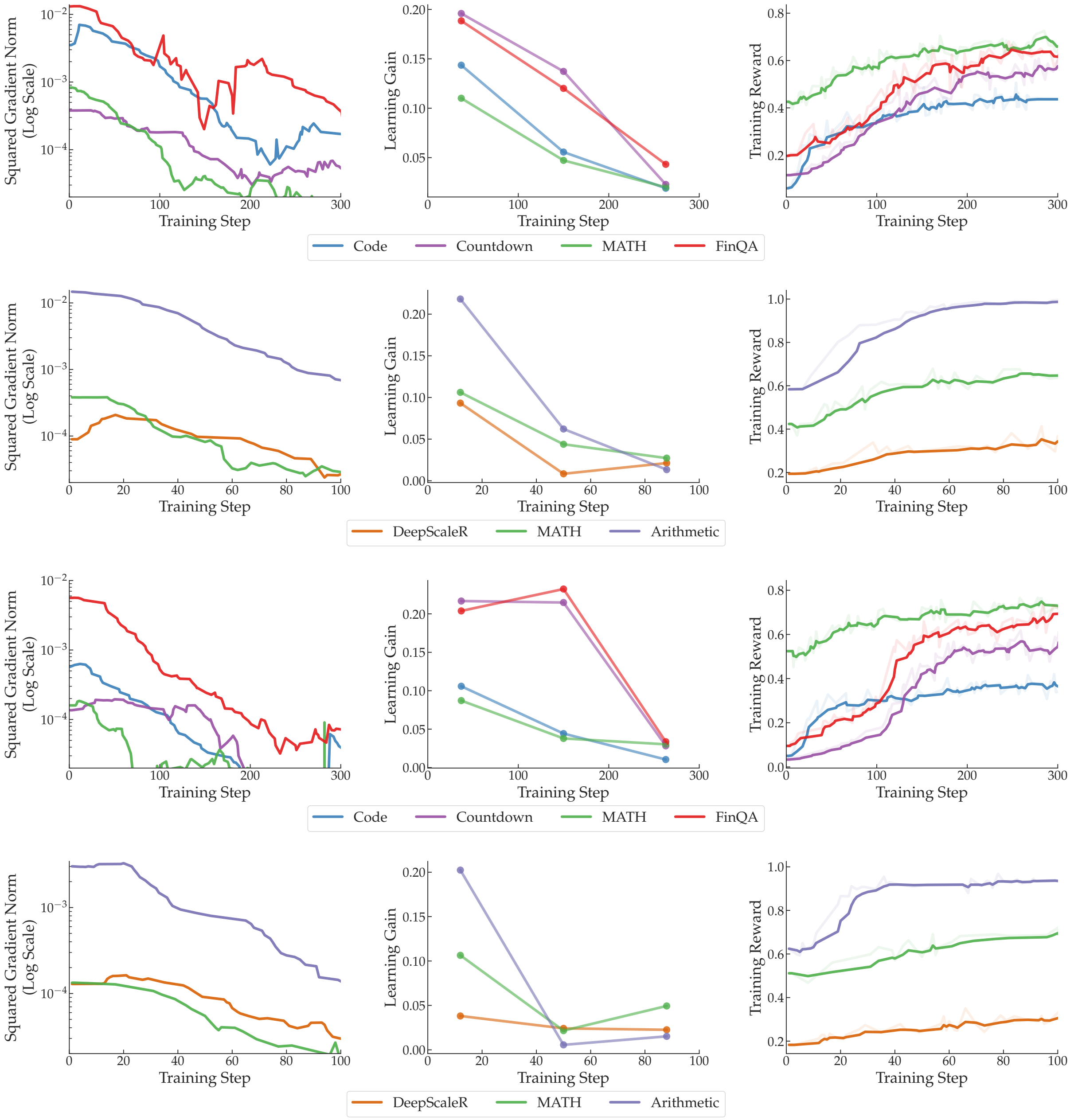

Figure 6: **Replication of gradient imbalance across models.** Same setup as Figure 1 but for Qwen-3B (top two rows) and Llama-3B (bottom two rows). The imbalance pattern persists.

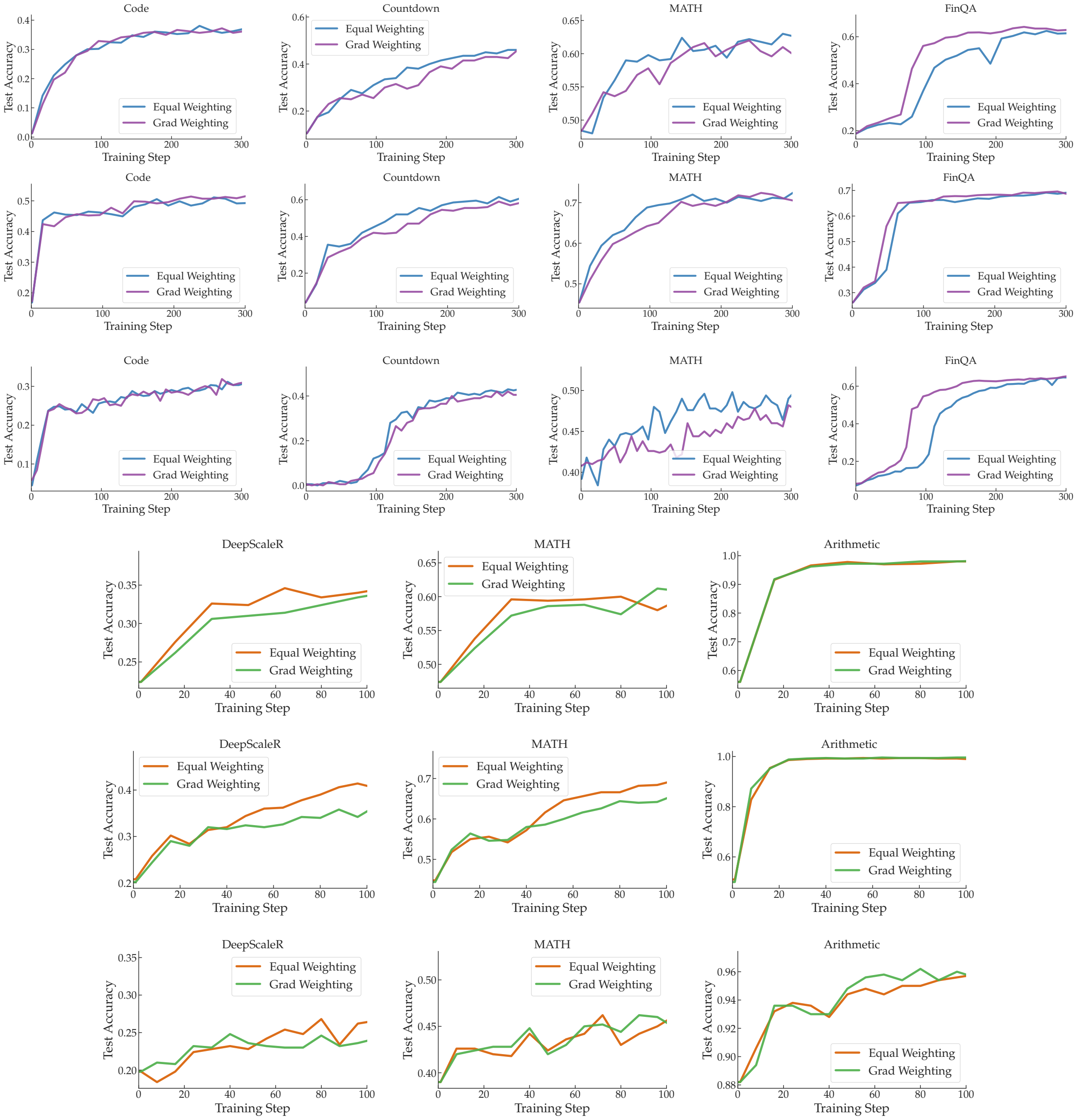

Figure 7: **Task-level accuracy under uniform vs. gradient-proportional sampling** (as reported in Table 1). Top three rows present multi-domain tasks, and bottom three present single-domain tasks. Within each block, results are shown for Qwen-3B, Qwen-7B, and Llama-3B from top to bottom.