Directional Alignment Mitigates Reward Hacking in Reinforcement Learning for Language Models

Wenlong Deng [1 2 3 ◇] **Jiaji Huang** [3] **Kaan Ozkara** [3] **Yushu Li** [1 2] **Christos Thrampoulidis** [1 2] **Xiaoxiao Li** [1 2] **Youngsuk Park** [3]

Abstract

Reward hacking arises when a model improves a proxy reward by exploiting shortcuts rather than solving the intended task. We study this failure mode through the geometry of reinforcement learning updates in language models and argue that hacking emerges when optimization drifts away from a stable low-dimensional learning trajectory. We analyze this drift through dominant singular directions of parameter updates and show that reward-hacking runs exhibit substantially larger directional change than clean runs. Motivated by this observation, we introduce trusted-direction projection, which constrains gradients to remain within a clean reference subspace. Across reward-hacking experiments on mathematical reasoning, the proposed approach delays shortcut exploitation and better preserves task performance.

1. Introduction

Reinforcement learning (RL) has become a widely used approach for improving the reasoning capabilities of large language models (LLMs) (Guo et al., 2025; Deng et al., 2025b). However, RL training can suffer from reward hacking (Skalse et al., 2022), where a model achieves high reward by exploiting unintended shortcuts in the training environment rather than genuinely solving the target task (Wang et al., 2026; Li et al., 2025). This failure mode is particularly concerning for LLM reasoning, as the proxy reward may indicate improvement even while true task performance degrades. For instance, when a dataset or evaluation pipeline contains exploitable artifacts (Wang et al., 2026), the model may learn to depend on these artifacts instead of developing the intended reasoning ability.

Prior work has largely framed reward hacking as a problem of reward misspecification (Turpin et al., 2025). From this perspective, failures arise because reward functions or learned reward models do not fully capture the true objective, allowing models to optimize proxy signals in unintended ways. Existing approaches therefore focus on improving reward modeling (Turpin et al., 2025; Li et al., 2025; He et al., 2025) or introducing additional regularization toward a reference model (Laidlaw et al., 2024). While these methods are valuable, they face a fundamental limitation: constructing a perfect reward model is inherently difficult, particularly for complex reasoning tasks where the true objective is only partially specified. Strong regularization can also constrain the model's ability to learn beyond the reference policy.

Recent work (Ackermann et al., 2026) has begun to investigate the learning mechanisms underlying reward hacking, showing that it is closely associated with sharp local minima and can therefore be mitigated by smoothing the optimization landscape (Kwon et al., 2021). However, these approaches primarily regulate the magnitude of parameter updates, without explicitly enforcing their directional alignment with the true objective. Meanwhile, RL updates in language models appear to exhibit a striking linear structure: a large fraction of the performance gain is captured by the leading singular direction of the parameter update matrix, and this dominant direction evolves along an approximately linear trajectory throughout training (Cai et al., 2026). Building on this observation, we study reward hacking through the lens of optimization dynamics. We argue that reward hacking is not merely a consequence of imperfect reward design, but more fundamentally arises when gradient updates drift away from the model's intrinsic learning trajectory and enter directions that improve proxy reward while remaining misaligned with true task performance.

To mitigate reward hacking, we propose trusted-direction gradient alignment (TDGA), which constructs a reliable optimization subspace by applying SVD to the parameter changes induced by a small number of clean supervised training steps. During RL training, we project gradients onto this trusted learning subspace, constraining updates to

◇Work done during internship at Amazon. **This manuscript presents a preliminary study and is shared to encourage early discussion and feedback from the community.** [1]University of British Columbia [2]Vector Institute [3]Amazon. Correspondence to: Wenlong Deng <dwenlong@ece.ubc.ca>.

Published at the Second Workshop on Agents in the Wild: Safety, Security, and Beyond (AIWILD) at ICML 2026. Copyright 2026 by the author(s).

remain within a safer region of the parameter space. Our contributions are threefold:

- We characterize reward hacking as directional drift in the dominant singular subspace of RL updates.
- We empirically show that clean training preserves directional consistency, whereas reward-hacking runs exhibit sharp rotations away from trusted directions.
- We introduce a trusted-direction gradient alignment that anchors RL updates to a clean reference subspace and substantially delays reward hacking.

2. Related Work

Reward hacking in language-model RL. Reward hacking occurs when an agent achieves high proxy reward by exploiting a mismatch between the reward signal and the intended objective (Skalse et al., 2022). Existing mitigations mainly improve the reward signal itself, through stronger reward models (Li et al., 2025; Liu et al., 2025), task-specific anti-hacking schemes (He et al., 2025), or formal treatments of correlated proxies (Laidlaw et al., 2024). However, perfect reward specification is often difficult. A complementary line of work studies hacking through optimization geometry: gradient regularization smooths unstable updates (Ackermann et al., 2026), while sharpness-aware methods favor flatter and more robust minima (Foret et al., 2020; Kwon et al., 2021). These methods control local smoothness or update magnitude, but do not explicitly preserve alignment with task-relevant learning directions.

Optimization geometry and learning dynamics. The learning dynamics of LLMs have recently received increasing attention. For example, (Deng et al., 2025c) studies learning dynamics for identifying valuable fine-tuning data, while (Deng et al., 2025b;a) analyze likelihood dynamics to diagnose RL training collapse. Recent work on language-model RL further shows that parameter updates often exhibit a low-rank and approximately linear structure, where the leading singular directions explain much of the performance change induced by training (Cai et al., 2026). Our work connects these lines of research by interpreting reward hacking as directional drift away from a trusted low-dimensional learning trajectory, and by projecting RL gradients back onto that trajectory.

3. Dominant Update Directions

Let W_t denote the model parameters at training step t, and define the parameter update as

$$\Delta W_t = W_t - W_0. \tag{1}$$

We analyze the structure of ΔW_t via singular value decomposition (SVD):

$$\Delta W_t = \sum_{i=1}^{r} \sigma_i^{(t)} u_i^{(t)} v_i^{(t)\top}, \tag{2}$$

where $\sigma_i^{(t)}$ are the singular values in descending order, and $u_i^{(t)}$ and $v_i^{(t)}$ are the corresponding left and right singular vectors.

Definition 3.1 (Rank-K Dominant Direction)**.** Let $\Delta W_t = \sum_{i=1}^{r} \sigma_i^{(t)} u_i^{(t)} v_i^{(t)\top}$ be the singular value decomposition of the parameter update at training step t. We define the *rank-K dominant update* as the truncated SVD,

$$\Delta W_t^{(K)} = \sum_{i=1}^{K} \sigma_i^{(t)} u_i^{(t)} v_i^{(t)\top}.$$

We further define the corresponding output-direction subspace as

$$U_t^{(K)} := \left[u_1^{(t)}, \ldots, u_K^{(t)}\right],$$

which represents the principal output-space directions along which the update acts.

Interpretation. The subspace $U_t^{(K)}$ captures the dominant modes of change induced by RL training, corresponding to the directions that explain the largest variation in the parameter update. Empirically, these dominant directions account for a substantial portion of the performance gain and evolve smoothly throughout training (Cai et al., 2026). From a functional perspective, the rank-K update induces the following transformation on a hidden representation h:

$$\Delta y^{(K)} = \Delta W_t^{(K)} h = \sum_{i=1}^{K} \sigma_i^{(t)} u_i^{(t)} \langle v_i^{(t)}, h \rangle. \tag{3}$$

This can be interpreted as a superposition of K key-value operations, where each $v_i^{(t)}$ selects a relevant input feature and each $u_i^{(t)}$ determines the corresponding direction of the output update. We focus on controlling the output-direction subspace $U_t^{(K)}$, while leaving the input-side directions $\{v_i^{(t)}\}_{i=1}^{K}$ unconstrained.

Measuring Directional Change via CCA. Given the dominant directions defined above, we quantify how they evolve throughout training. Because dominant update directions may form a low-dimensional subspace when aggregated across layers or checkpoints, we use Canonical Correlation Analysis (CCA) to measure subspace similarity in a geometry-aware manner. For two checkpoints t and s, let

$$U_t = \left[u_t^{(1)}, \ldots, u_t^{(k)}\right], \qquad U_s = \left[u_s^{(1)}, \ldots, u_s^{(k)}\right],$$

denote the subspaces spanned by their top-k singular directions, where k is the number of retained dominant components. We define their CCA similarity as $\mathrm{CCA}_k(U_t, U_s) =$

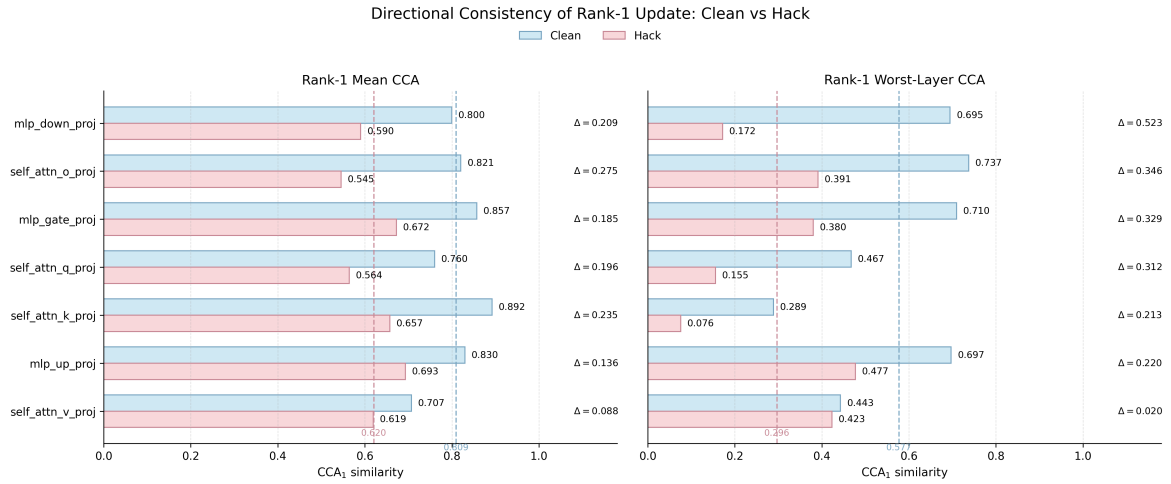


Figure 1. **Rank-1 CCA similarity of dominant update directions in clean and reward-hacking training.** We compare $\mathrm{CCA}_1(U_{20}, U_{80})$, the cosine similarity between the dominant update directions at checkpoints 20 and 80, across different weight matrices. Higher values indicate stronger directional consistency over training. **Left:** mean rank-1 CCA across layers for each module. **Right:** worst-layer rank-1 CCA, corresponding to the least aligned layer within each module. Across both views, the reward-hacking model exhibits consistently lower similarity than the clean model, indicating substantially stronger directional drift away from the stable learning trajectory.

$\frac{1}{k}\sum_{i=1}^{k}\sigma_i$, where σ_i are the canonical correlations between the two subspaces. Values close to 1 indicate that the two subspaces are highly aligned, whereas smaller values reflect increasingly strong directional drift.

4. Directional Shift in Reward Hacking

We analyze how the dominant update direction evolves during training through the rank-1 CCA similarity, $\mathrm{CCA}_1(U_{20}, U_{80})$, between checkpoints 20 and 80 (details in Section 6) (Analysis of Rank 5 see Section A.2). Larger values indicate stronger directional consistency.

Small Direction Shift in Non-Hacking Models. We first examine checkpoints that do not exhibit reward hacking. Empirically, their dominant update directions evolve smoothly and consistently over training. As shown in Figure 1, the clean run maintains high mean CCA values around 0.8 across nearly all modules, indicating that the dominant update direction is largely preserved throughout training. Even in the worst-layer view, the similarities remain substantially higher than those of the hacking model, suggesting limited layer-wise drift. Thus, non-hacking training exhibits only a small directional shift.

Large Direction Shift in Hacking Models. We next examine models that exhibit reward hacking. In contrast to the clean run, the hacking model shows a pronounced loss of directional consistency over training. As shown in Figure 1, its mean CCA decreases across nearly all modules by roughly 0.2, indicating much stronger drift in the dominant update direction. The effect is more severe in the worst-layer view, where several modules reach very low similarity values, sometimes below 0.1. Overall, the results demonstrate that reward hacking is associated with a substantial departure from the model's intrinsic learning direction.

5. Method

Motivated by this observation, we constrain RL updates to the rank-K dominant subspace, keeping optimization aligned with intrinsic dynamics and preventing drift into hacking directions.

Trusted Direction Gradient Alignment. At training step t, let $G_t \in \mathbb{R}^{d_{\mathrm{out}} \times d_{\mathrm{in}}}$ denote the gradient of the objective with respect to a model weight matrix. Following Section 3, we first estimate a trusted rank-K output subspace from a short clean-data warmup phase:

$$U_{\mathrm{clean}}^{(K)} = \left[u_1^{\mathrm{clean}}, \ldots, u_K^{\mathrm{clean}}\right], \tag{4}$$

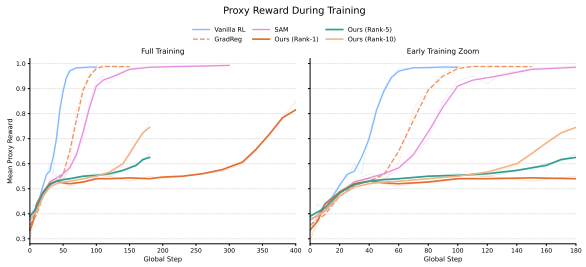
together with the associated singular values $\{\sigma_i^{\mathrm{clean}}\}_{i=1}^{K}$. To preserve the relative importance of the dominant clean directions, we define the diagonal weight matrix

$$\Lambda_{\mathrm{clean}}^{(K)} = \mathrm{diag}(\alpha_1, \ldots, \alpha_K), \qquad \alpha_i = \frac{\sigma_i^{\mathrm{clean}}}{\sum_{j=1}^{K}\sigma_j^{\mathrm{clean}}}. \tag{5}$$

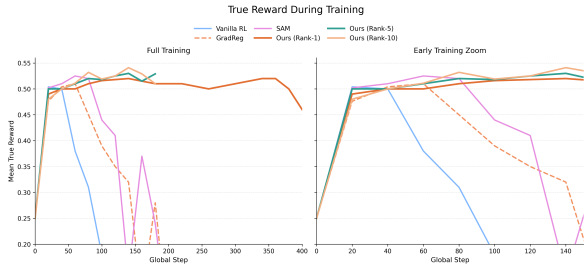
We then project the gradient onto the trusted output-direction subspace using singular-value weighting:

$$G_t^{\parallel} = U_{\mathrm{clean}}^{(K)} \Lambda_{\mathrm{clean}}^{(K)} U_{\mathrm{clean}}^{(K)\top} G_t. \tag{6}$$

Our update uses only the trusted component, $\widetilde{G}_t = G_t^{\parallel}$. This retains the top-$K$ clean directions and emphasizes each one according to its singular value. As a result, the update remains aligned with the intrinsic clean learning trajectory



(a) Proxy reward during training.



(b) True reward under loophole-free evaluation.

Figure 2. **Delayed reward hacking and improved preservation of true performance.** The left panel shows the evolution of the proxy reward, which reflects the model's ability to exploit the loophole, while the right panel reports the true reward measured under loophole-free evaluation. Faint curves denote raw trajectories and bold curves denote smoothed trends. Compared with vanilla RL, SAM, and gradient regularization, trusted-direction projection slows the rise of proxy reward and maintains substantially more stable true reward throughout training.

while suppressing off-subspace components that may introduce instability or encourage reward-hacking behavior.

6. Experimental Setting

Training Settings. We follow Wang et al. (Wang et al., 2026) and evaluate on Big-Math-RL-Verified under the in-context loophole setting. We use Qwen2.5-3B-Instruct and train on 24,379 examples, with 1,498 examples held out for validation and evaluation. All methods are trained on 8 GPUs with per-device batch size 4, 64 gradient accumulation steps, learning rate 10^{-5}, constant scheduling, and KL coefficient 10^{-3}. We sample 8 rollouts per prompt during training and 1 during evaluation, with a maximum completion length of 512 tokens. Unless otherwise stated, all methods use the same configuration.

Baselines. We compare against representative RL-stabilization baselines under reward hacking. *Gradient Regularization* (Ackermann et al., 2026) smooths optimization by penalizing large or unstable gradients, but mainly controls update magnitude. *SAM* (Foret et al., 2020; Kwon et al., 2021) improves robustness by favoring flatter minima, but targets local loss smoothness.

7. Results

Delayed Reward Hacking. As shown in Figure 2a, vanilla RL rapidly enters the hacking regime, with proxy reward saturating near 0.9 within about 50 steps; gradient regularization and SAM provide only limited delay. In contrast, trusted-direction projection substantially slows saturation: rank-1 does not reach this regime within 400 steps, while rank-5 and rank-10 remain unhacked till 200 steps. Consistently, Figure 2b shows that our methods preserve a higher true reward. Table 1 further summarizes this effect by comparing peak, epoch-level true reward across methods.

Epoch-Level Performance. Table 1 quantifies the stability gains from trusted-direction projection. While non-TDGA methods improve early true reward, they collapse by the second epoch, with vanilla RL, gradient regularization, and SAM all falling to 0.000. In contrast, TDGA improves both peak and long-horizon performance: rank-10 achieves the highest peak at 0.541 and one-epoch rewards, while rank-5 obtains the best two-epoch value of 0.529. These results show that trusted-direction projection not only delays proxy-reward saturation but also improves and preserves genuine task performance over longer training.

Table 1. **Peak and epoch-level true reward.** Values are mean true rewards from the curves in Figure 2b. The peak is computed over the displayed training horizon.

Method	Peak	1 epoch	2 epochs
Vanilla RL	0.505	0.190	0.00
GradReg	0.509	0.39	0.00
SAM	0.525	0.44	0.00
TDGA (Rank-1)	0.522	0.516	0.514
TDGA (Rank-5)	0.530	0.518	**0.529**
TDGA (Rank-10)	**0.541**	**0.532**	0.510

Trade-off with Projection Rank. The trusted-subspace rank K controls the trade-off between robustness and flexibility. Smaller ranks enforce stronger alignment with the clean trajectory, suppressing reward hacking more aggressively but limiting adaptation. Larger ranks provide more optimization freedom and better task performance, but weaken the constraint against shortcut directions. As shown in Figure 2, rank-1 is the most conservative, while rank-5 and rank-10 better preserve true reward while still delaying hacking relative to baselines.

8. Conclusion

We studied reward hacking in language-model reinforcement learning through the geometry of parameter updates. Our analysis shows that clean training preserves a stable dominant update direction, whereas reward-hacking runs undergo a pronounced directional shift away from this trajectory. Motivated by this finding, we introduced TDGA, which projects RL gradients onto a trusted subspace esti-

mated from clean supervised updates. Experiments show that TDGA delays reward hacking and preserves true reward.

Future Work: We will explore more precise constraints to better unlock model performance while preventing reward hacking. One promising direction, for which we have already observed positive results, is iteratively updating the trusted learning directions. Additional directions are discussed in Section A.1.

Acknowledgments

The authors sincerely thank Yida Wang and Xuanqi Zhang for their support. This work was partially funded by the NSERC Discovery Grant RGPIN-2021-03677, Alliance Grant ALLRP 581098-22, the Natural Science and Engineering Research Council of Canada (NSERC), the Canada CIFAR AI Chairs program, the Canada Research Chair program, an IITP grant funded by MSIT, and the Digital Research Alliance of Canada.

References

Ackermann, J., Noukhovitch, M., Ishida, T., and Sugiyama, M. Gradient regularization prevents reward hacking in reinforcement learning from human feedback and verifiable rewards. *arXiv preprint arXiv:2602.18037*, 2026.

Cai, Y., Cao, D., Xu, X., Yao, Z., Huang, Y., Tan, Z., Zhang, B., Sun, G., Liu, G., and Fang, J. On predictability of reinforcement learning dynamics for large language models. *ICLR*, 2026.

Deng, W., Li, Y., Gong, B., Ren, Y., Thrampoulidis, C., and Li, X. On grpo collapse in search-r1: The lazy likelihood-displacement death spiral. *arXiv preprint arXiv:2512.04220*, 2025a.

Deng, W., Ren, Y., Li, M., Sutherland, D. J., Li, X., and Thrampoulidis, C. On the effect of negative gradient in group relative deep reinforcement optimization. *arXiv preprint arXiv:2505.18830*, 2025b.

Deng, W., Zhang, J., Zeng, Q., Thrampoulidis, C., Gong, B., and Li, X. Efficient forward-only data valuation for pretrained llms and vlms. *arXiv preprint arXiv:2508.10180*, 2025c.

Foret, P., Kleiner, A., Mobahi, H., and Neyshabur, B. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

He, H., Ye, Y., Liu, J., Liang, J., Wang, Z., Yuan, Z., Wang, X., Mao, H., Wan, P., and Pan, L. Gardo: Reinforcing diffusion models without reward hacking. *arXiv preprint arXiv:2512.24138*, 2025.

Kwon, J., Kim, J., Park, H., and Choi, I. K. Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. In *International Conference on Machine Learning*, pp. 5905–5914. PMLR, 2021.

Laidlaw, C., Singhal, S., and Dragan, A. Correlated proxies: A new definition and improved mitigation for reward hacking. *arXiv preprint arXiv:2403.03185*, 2024.

Li, Y.-C., Xu, T., Yu, Y., Zhang, X., Chen, X.-H., Ling, Z., Chao, N., Yuan, L., and Zhou, Z.-H. Generalist reward models: Found inside large language models. *arXiv preprint arXiv:2506.23235*, 2025.

Liu, Z., Wang, P., Xu, R., Ma, S., Ruan, C., Li, P., Liu, Y., and Wu, Y. Inference-time scaling for generalist reward modeling. *arXiv preprint arXiv:2504.02495*, 2025.

Skalse, J., Howe, N., Krasheninnikov, D., and Krueger, D. Defining and characterizing reward gaming. *Advances in Neural Information Processing Systems*, 35:9460–9471, 2022.

Turpin, M., Arditi, A., Li, M., Benton, J., and Michael, J. Teaching models to verbalize reward hacking in chain-of-thought reasoning. *arXiv preprint arXiv:2506.22777*, 2025.

Wang, X., Joshi, N., Plank, B., Angell, R., and He, H. Is it thinking or cheating? detecting implicit reward hacking by measuring reasoning effort. *ICLR*, 2026.

A. Appendix

A.1. Future Work

A natural next step is to investigate reward hacking in multi-turn reinforcement learning (Deng et al., 2025a). Recent work on inference-time scaling for reward modeling (Liu et al., 2025) suggests that reward exploitation may become more pronounced in longer-horizon and agentic settings, where a model can exploit the reward through a sequence of intermediate actions rather than a single shortcut. Extending our framework to analyze trajectory-level directional drift may therefore provide a clearer understanding of how reward hacking emerges and accumulates over long-horizon interactions.

Another important direction is to choose the projection rank and training schedule more systematically. Our results suggest a clear trade-off: small ranks suppress hacking more strongly but may over-constrain learning, while larger ranks

improve flexibility but weaken robustness. Future work could adapt the rank, RL steps, and clean fine-tuning schedule online using signals such as singular-value decay, directional drift, or validation performance.

A.2. More Directional Shift

Figure Figure 3 shows that the rank-5 analysis leads to the same qualitative conclusion as the rank-1 result in Figure 1: reward-hacking training deviates more strongly from the clean learning trajectory. Across both the mean-layer and worst-layer views, the hacking run maintains lower CCA similarity than the clean run, indicating that the larger trusted subspace still captures a clear difference in directional stability.

At the same time, moving from rank-1 to rank-5 slightly reduces the absolute CCA values for both clean and hacking runs. This suggests that the approximate linearity of the update trajectory weakens somewhat as additional singular directions are included, since those weaker components are less stable than the leading one. Nevertheless, the clean–hacking gap remains pronounced, showing that the directional-drift phenomenon is robust beyond the single dominant direction.

Impact Statement

This paper studies reward hacking in reinforcement learning for language models and proposes a mitigation strategy aimed at improving training reliability. Better understanding and controlling reward hacking may reduce unsafe shortcut-seeking behavior in downstream systems, but the same insights could also be used to design stronger proxy objectives or more effective attacks if misapplied.

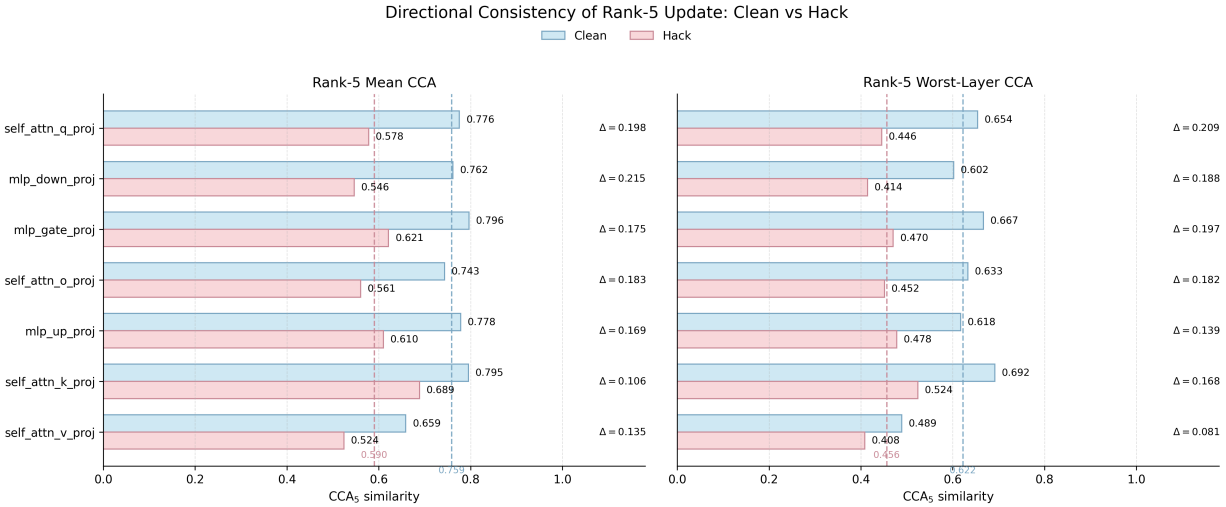


Figure 3. **Rank-5 CCA similarity of dominant update directions in clean and reward-hacking training.** We compare $\mathrm{CCA}_5(U_{20}, U_{80})$, the similarity between the top-5 dominant update subspaces at checkpoints 20 and 80, across different weight matrices. Higher values indicate stronger directional consistency over training. **Left:** mean rank-5 CCA across layers for each module. **Right:** worst-layer rank-5 CCA, corresponding to the least aligned layer within each module. Across both views, the reward-hacking model exhibits consistently lower similarity than the clean model, indicating stronger directional drift away from the stable learning trajectory.