大模型推理强化学习的熵机制

The Entropy Mechanism of Reinforcement Learning for Reasoning Language Models

Paper

http://arxiv.org/abs/2505.22617

Code

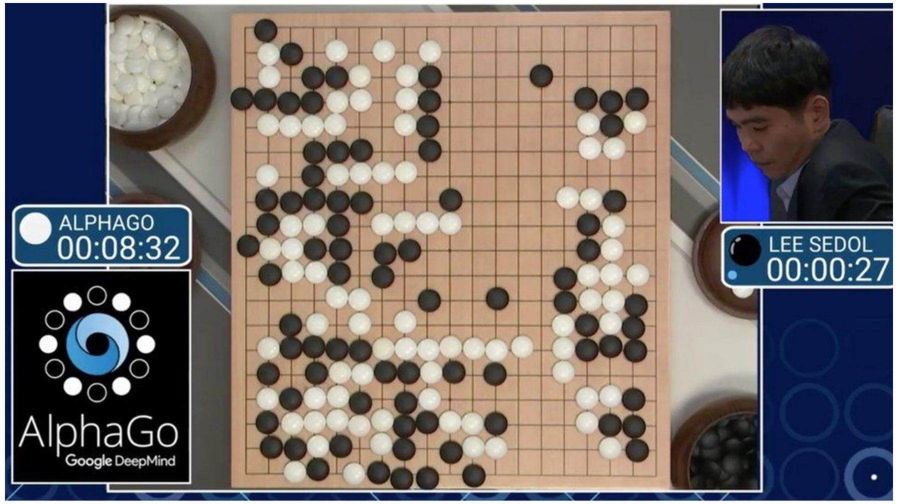
Ganqu Cui, Yuchen Zhang, Jiacheng Chen and Ning Ding et.al

2025.07.01

上海人工智能实验室
Shanghai Artificial Intelligence Laboratory

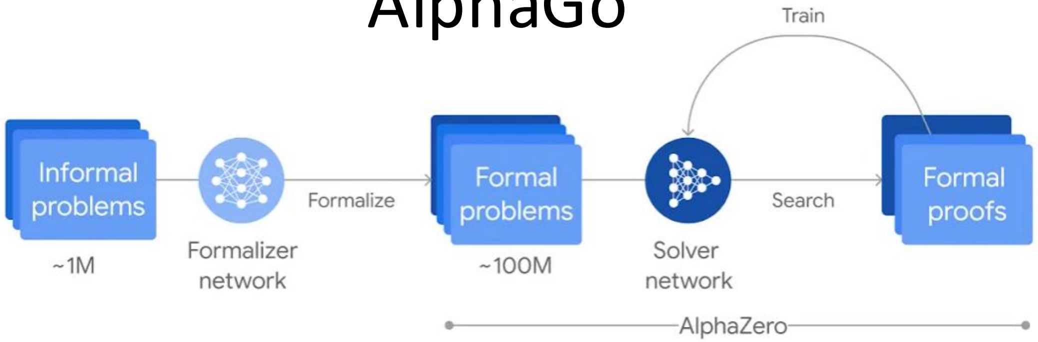
Why Reinforcement Learning

Some of the AI breakthroughs in the past **10 years**

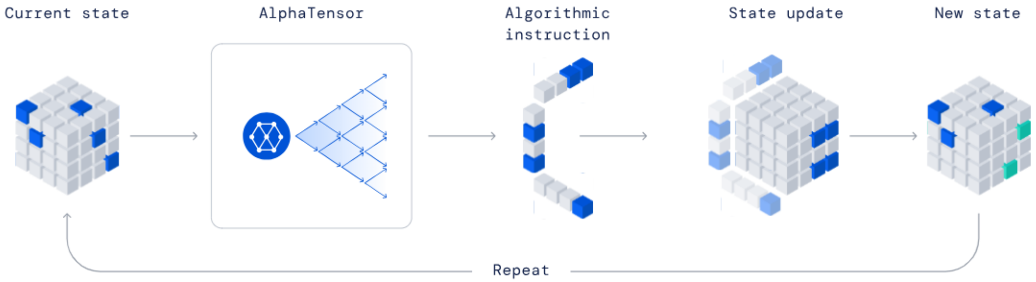


AlphaGo

AlphaStar



AlphaProof



AlphaTensor

Why Reinforcement Learning

Some of the AI breakthroughs in the past **1 year**

Our large-scale reinforcement learning algorithm teaches the model how to think productively using its chain of thought in a highly data-efficient training process. We have found that the performance of o1 consistently improves with more reinforcement learning (train-time compute) and with more time spent thinking (test-time compute). The

OpenAI o1

deepseek

DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning

DeepSeek-AI

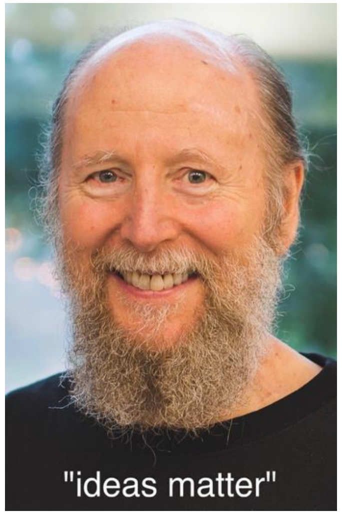
research@deepseek.com

DeepSeek R1

Why Reinforcement Learning

The next Scaling Law?

One thing that should be learned from the bitter lesson is the great power of general purpose methods, of methods that continue to **scale with increased computation** even as the available computation becomes very great. The two methods that seem to scale arbitrarily in this way are ***search* and *learning***.



Richard Sutton
(ACM Turing Award)
The Bitter Lesson

Reinforcement learning　　Pretraining and finetuning

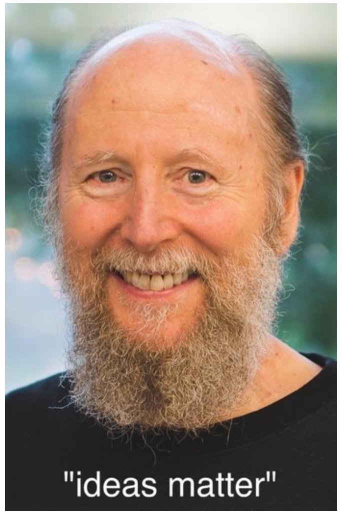
Why Reinforcement Learning

Welcome to the Era of Experience

David Silver, Richard S. Sutton*

Consequently, the methodology of experiential RL was largely discarded in favour of more general-purpose agents, resulting in a widespread transition to human-centric AI. However, something was lost in this transition: an agent's ability to ***self-discover its own knowledge***. The era of experience will reconcile this ability with the level of task generality achieved in the era of human data.

David Silver
AlphaGo, AlphaZero



Richard Sutton
(ACM Turing Award)

Why *haven't* Reinforcement Learning

RL with LLMs haven't been scaled well

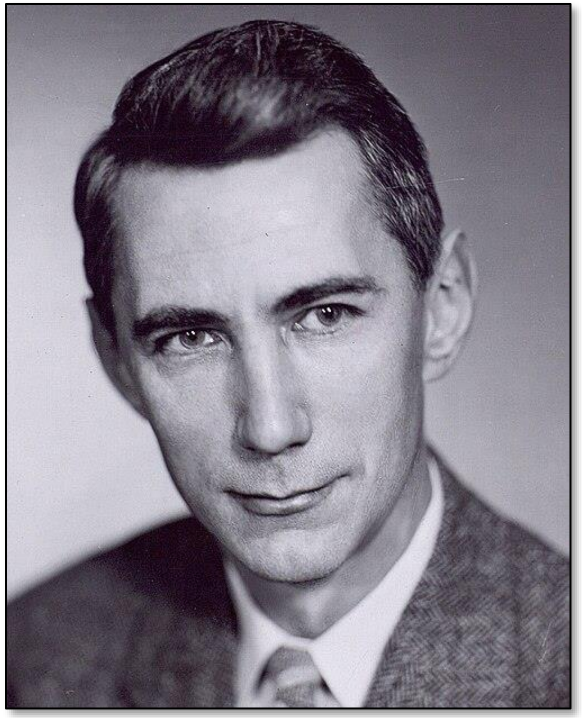
- Most open-source models can only be trained for several hundred steps
- The training compute in RL is still smaller than pre-training in magnitude
- ***Why can't we train LLMs with RL for months?***

A Major Obstacle: *Entropy Collapse*

What is Entropy?

- a concept commonly associated with states of *disorder, randomness, or uncertainty*
- Stemmed from Thermodynamics
- Introduced in Information Theory by Claude Shannon

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log p(x_i)$$

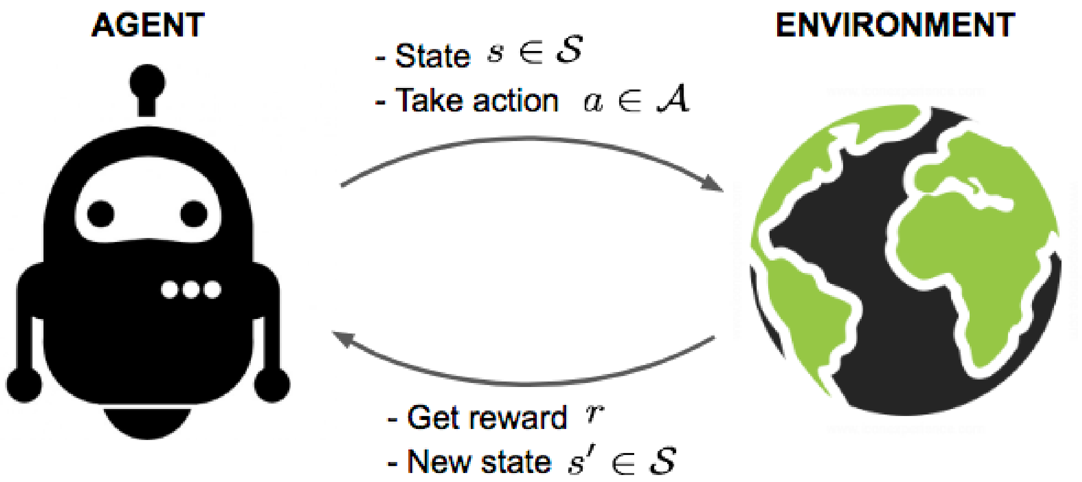


Claude Shannon

A Major Obstacle: *Entropy Collapse*

Entropy is basic in reinforcement learning

- RL is about *exploration-exploitation* tradeoff
- Entropy is a good measure of exploration
- Widely-adopted regularization term (maximum entropy RL)



https://lilianweng.github.io/posts/2018-02-19-rl-overview/

A Major Obstacle: *Entropy Collapse*

However, in RL for LLMs

- Entropy regularization is rarely considered
- Typically, we find that
  - policy entropy encounters a *sharp drop*
  - performance rises rapidly, then *saturates*

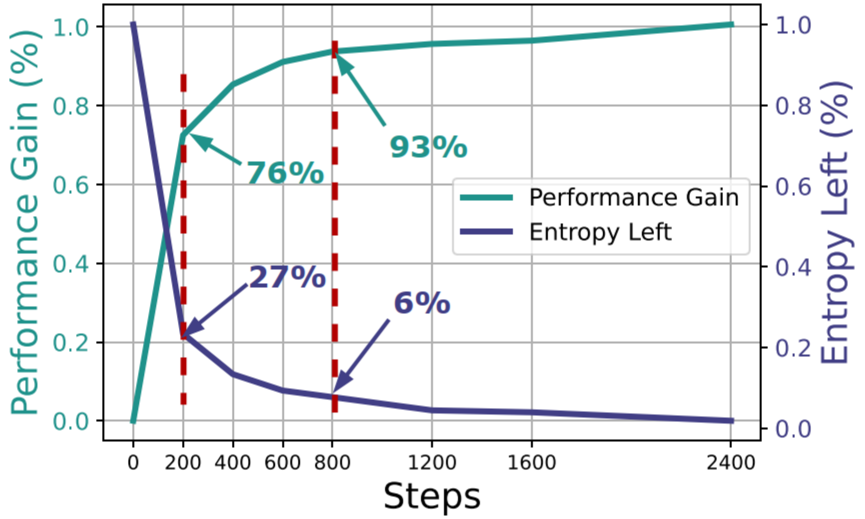
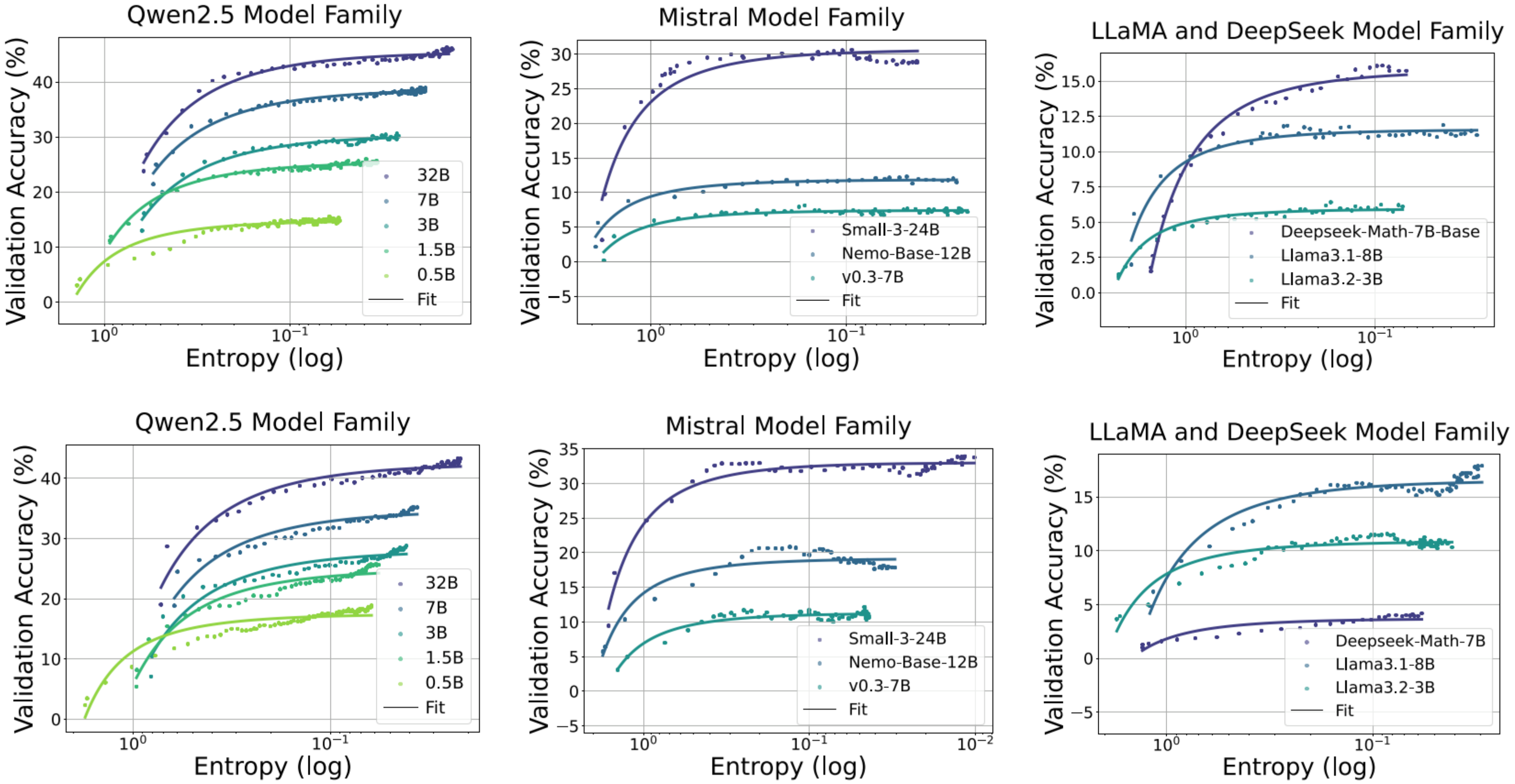


Figure 2: Avg. entropy consumption/performance gain (%) from 11 RL runs with different models.

A Major Obstacle: *Entropy Collapse*

What if we put them together?

- A strong correlation between policy entropy and performance



A Major Obstacle: *Entropy Collapse*

What if we put them together?

- A strong correlation between policy entropy and performance
- We get an **empirical function** to describe it

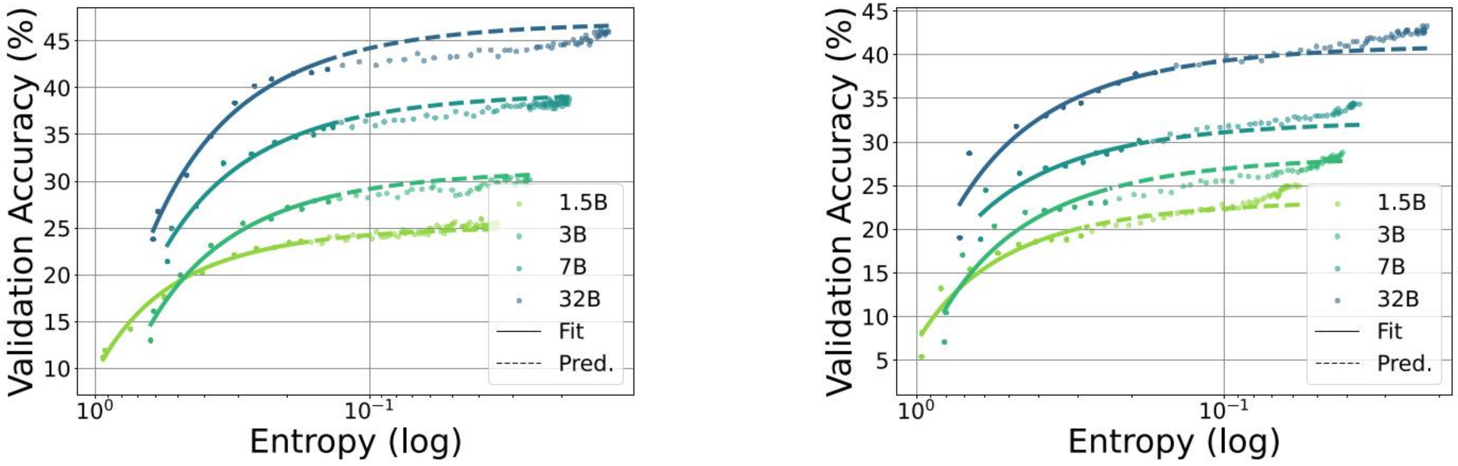
$$R = -a \exp(\mathcal{H}) + b$$

- It means that, we can **predict policy performance from its entropy**

A Major Obstacle: *Entropy Collapse*

What does this function implicate? $R = -a\exp(\mathcal{H}) + b$

- Predicting late stage from early stage



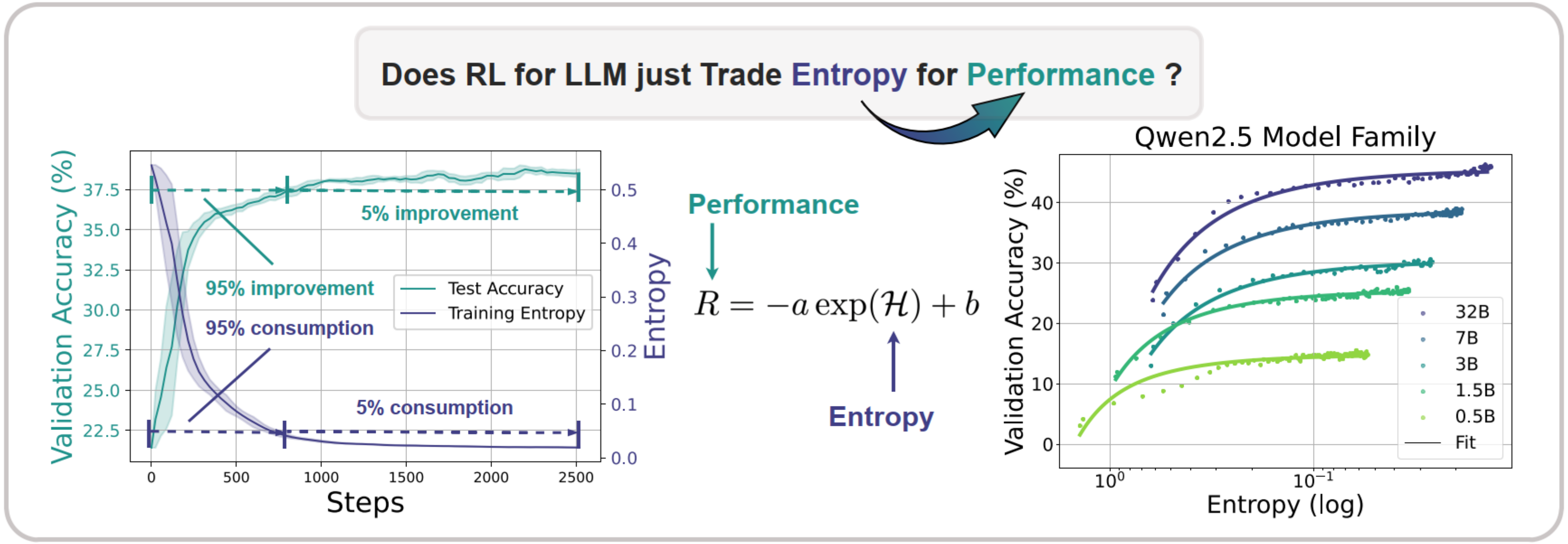
(a) Math Task (b) Code Task

Figure 5: Predicting the final performance of Qwen2.5 family with only 15% training steps with the fitted function. The average RMSE is 0.9% and 1.2% for all predicted steps, 0.5% and 1.9% for final step performance, respectively.

A Major Obstacle: *Entropy Collapse*

What does this function implicate? $R = -a\exp(\mathcal{H}) + b$

- *Without entropy intervention, RL is just trading entropy for performance*
- *The ceiling of RL is pre-determined* $(\mathcal{H} = 0, R = -a + b)$



A Major Obstacle: *Entropy Collapse*

What affect the coefficients? $R = -a\exp(\mathcal{H}) + b$

- The coefficients are algorithm-irrelevant

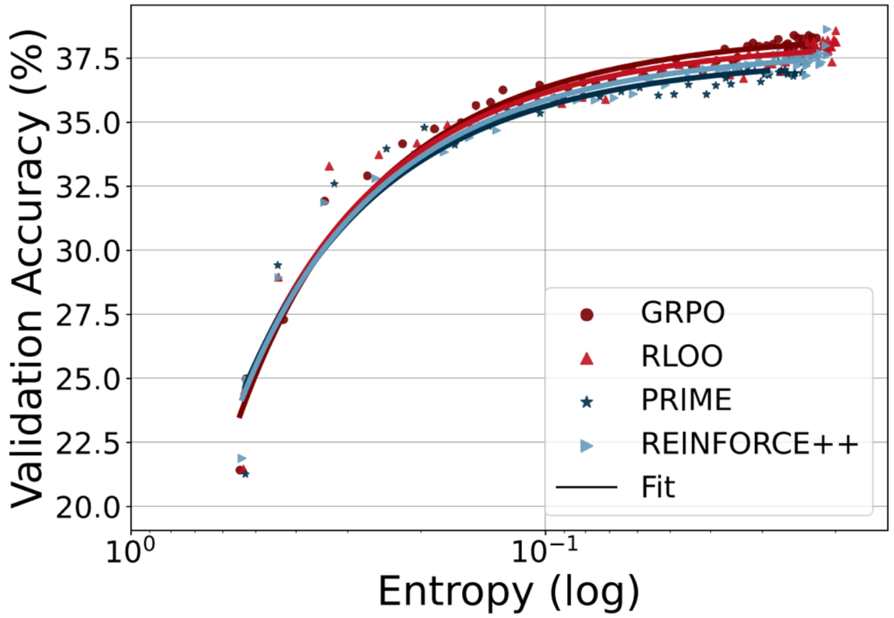
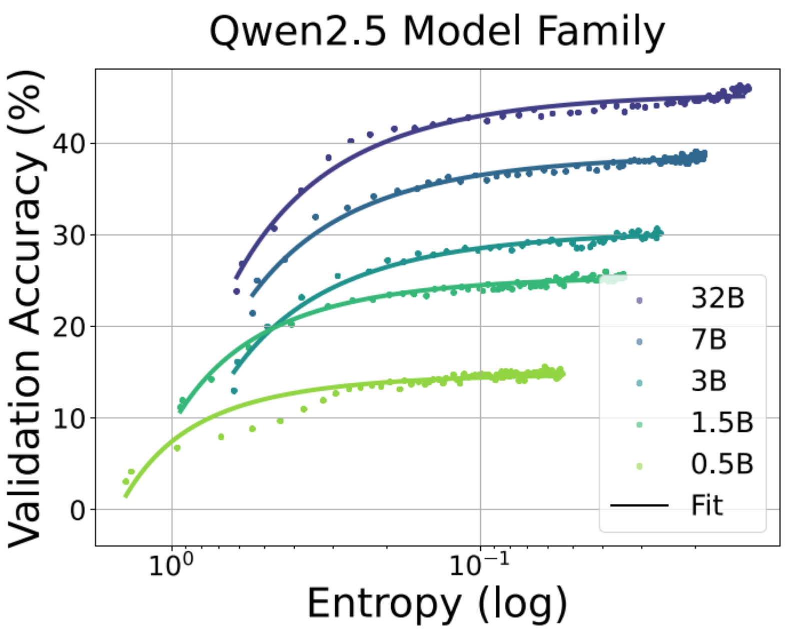


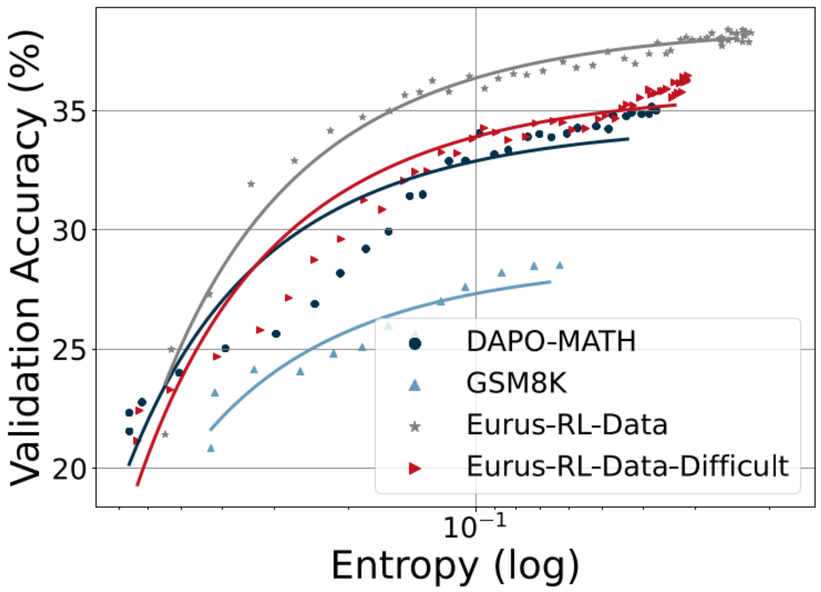
Figure 6: Training Qwen2.5-7B with different RL algorithms.

A Major Obstacle: *Entropy Collapse*

What affect the coefficients? $$R = -a \exp(\mathcal{H}) + b$$

- The coefficients are algorithm-irrelevant
- The policy model and training data are relevant

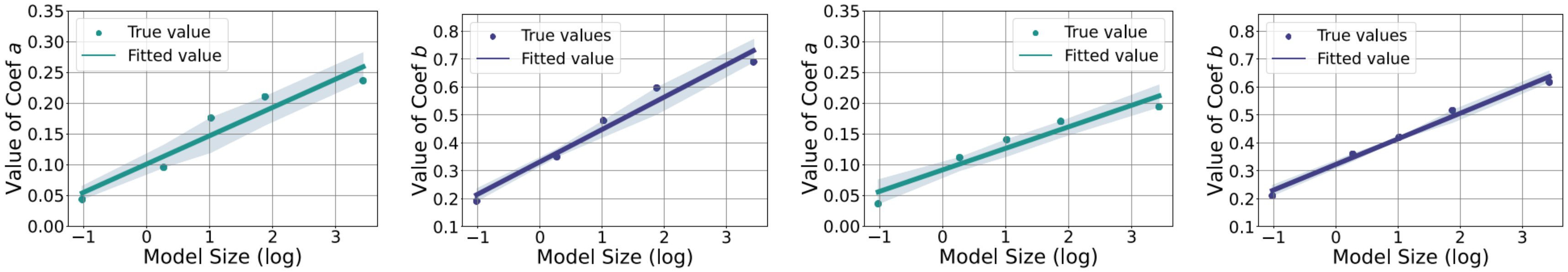




A Major Obstacle: *Entropy Collapse*

What affect the coefficients? $R = -a\exp(\mathcal{H}) + b$

- The coefficients are algorithm-irrelevant
- The policy model and training data are relevant
- *We can even predict the coefficients of large models from small models*



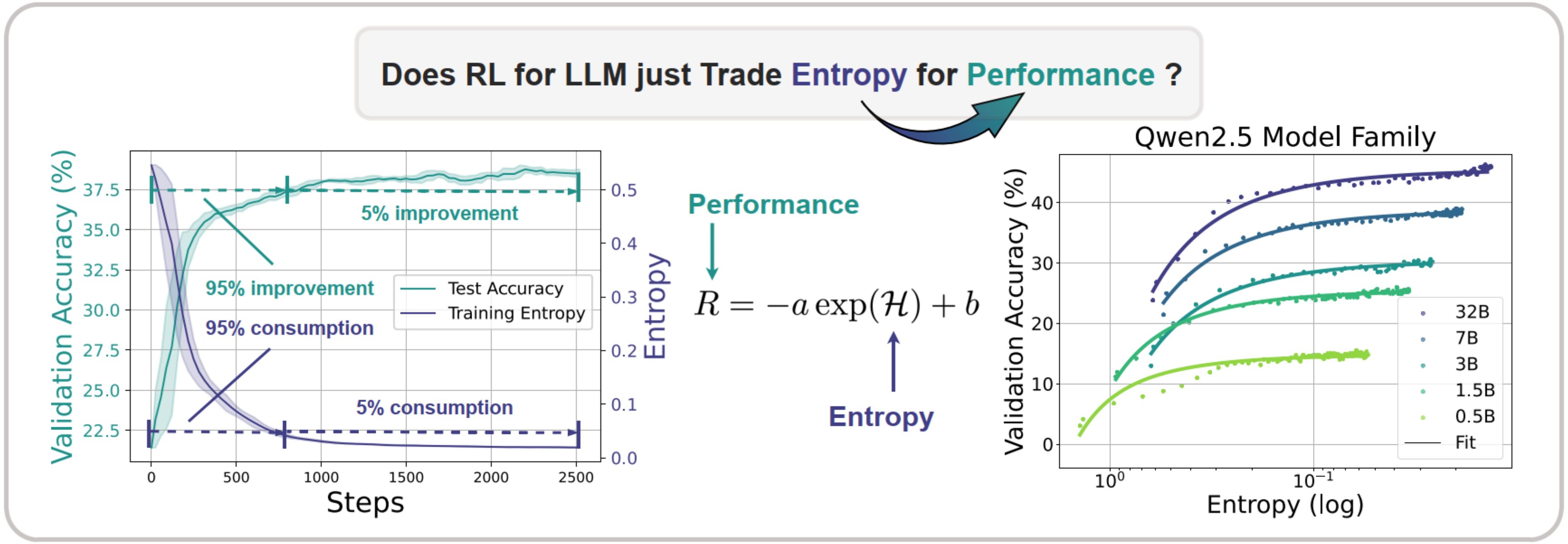
(a) Coef. a for math task (b) Coef. b for math task (c) Coef. a for code task (d) Coef. b for code task

Figure 7: Fitted curves between coefficients and model sizes of Qwen2.5 model family. The model sizes are parameter counts (B) without embeddings. a, b are obtained from experiments in Sec. 2.4. We use log-linear function to fit the curve.

A Major Obstacle: *Entropy Collapse*

The observation seems pessimistic 🙁

- *The ceiling not only exists, but also is predictable*
- *Does RL merely elicit the latent behaviors in the base model?*



A Major Obstacle: *Entropy Collapse*

TAKEAWAY

Without intervention, e.g., entropy or KL regularization, policy entropy is ***traded for reward predictably*** during RL. The empirical quantitative relationship between validation reward R and policy entropy $\mathcal{H}$ can be expressed as $R = -a\exp(\mathcal{H} + b)$. Then:

- It suggests the predictability of policy performance from entropy;
- The coefficients a, b reflect internal characteristics of policy and data;
- The performance ceiling of the policy ($\mathcal{H} = 0$, $R = -a + b$) greatly burdens the scalability of RL for LLM reasoning.

Understanding Entropy Dynamics

We want to break the ceiling

- So we need to understand the ***dynamics*** of policy entropy
- At each step, what makes entropy decrease and what makes it increase?
- Analyze step-wise entropy difference

$$\mathcal{H}(\pi_\theta^{k+1}) - \mathcal{H}(\pi_\theta^{k})$$

Understanding Entropy Dynamics

Entropy Dynamics of **Softmax Policy**

- LLMs are Softmax policies $\pi_\theta(a|s) = \frac{\exp(z_{s,a})}{\sum_{a' \in \mathcal{A}} \exp(z_{s,a'})}.$
- Proportional to the *covariance* of log-probability and logits difference

Lemma 1 (Entropy difference of softmax policy) *(Proof in Appendix B.2, adapted from Liu (2025)) Assume that policy π_θ is a tabular softmax policy, where each state-action pair (s, a) is associated with an individual logit parameter $z_{s,a} = \theta_{s,a}$, the difference of policy entropy given state s between two consecutive steps under first-order approximation satisfies*

$$\mathcal{H}(\pi_\theta^{k+1}) - \mathcal{H}(\pi_\theta^k) \approx \mathbb{E}_{s \sim d_{\pi_\theta}} \left[\mathcal{H}(\pi_\theta^{k+1}|s) - \mathcal{H}(\pi_\theta^k|s)\right] \approx \mathbb{E}_{s \sim d_{\pi_\theta}} \left[-Cov_{a \sim \pi_\theta^k(\cdot|s)} \left(\log \pi_\theta^k(a|s),\ z_{s,a}^{k+1} - z_{s,a}^k\right)\right]$$

Jiacai Liu, https://zhuanlan.zhihu.com/p/28476703733

Understanding Entropy Dynamics

Entropy Dynamics of **PG/NPG**

- For PG-like algorithms

Theorem 1 (Entropy change under policy gradient) *Let the actor policy π_θ be a tabular softmax policy, and π_θ be updated via vanilla policy gradient, the difference of policy entropy given state s between two consecutive steps satisfies*

$$\mathcal{H}(\pi_\theta^{k+1}|s) - \mathcal{H}(\pi_\theta^{k}|s) \approx -\eta \cdot Cov_{a\sim\pi_\theta^k(\cdot|s)}\left(\log \pi_\theta^k(a|s)\,,\pi_\theta^k(a|s)\cdot A(s,a)\right)$$

- For NPG-like algorithms

Theorem 2 (Entropy change under natural policy gradient) *(Proof in Appendix B.4) Let the actor policy π_θ be a tabular softmax policy, and π_θ is updated via natural policy gradient (Kakade, 2001), the difference of policy entropy given state s between two consecutive steps satisfies*

$$\mathcal{H}(\pi_\theta^{k+1}|s) - \mathcal{H}(\pi_\theta^{k}|s) \approx -\eta \cdot Cov_{a\sim\pi_\theta^k(\cdot|s)}\left(\log \pi_\theta^k(a|s)\,, A(s,a)\right)$$

Jiacai Liu, https://zhuanlan.zhihu.com/p/28476703733

Understanding Entropy Dynamics

Entropy Dynamics of **PG/NPG**

- For PG/NPG, logits change is proportional to ***action advantage***

Theorem 1 (Entropy change under policy gradient) *Let the actor policy π_θ be a tabular softmax policy, and π_θ be updated via vanilla policy gradient, the difference of policy entropy given state s between two consecutive steps satisfies*

$$\mathcal{H}(\pi_\theta^{k+1}|s) - \mathcal{H}(\pi_\theta^{k}|s) \approx -\eta \cdot Cov_{a\sim\pi_\theta^k(\cdot|s)}\left(\log \pi_\theta^k(a|s)\ , \pi_\theta^k(a|s) \cdot A(s,a)\right)$$

Theorem 2 (Entropy change under natural policy gradient) *(Proof in Appendix B.4) Let the actor policy π_θ be a tabular softmax policy, and π_θ is updated via natural policy gradient (Kakade, 2001), the difference of policy entropy given state s between two consecutive steps satisfies*

$$\mathcal{H}(\pi_\theta^{k+1}|s) - \mathcal{H}(\pi_\theta^{k}|s) \approx -\eta \cdot Cov_{a\sim\pi_\theta^k(\cdot|s)}\left(\log \pi_\theta^k(a|s)\ , A(s,a)\right)$$

Understanding Entropy Dynamics

Empirical Verification

- Under on-policy PG

$$Cov_{a\sim\pi_\theta(\cdot|s)}\left(\log\pi_\theta(a\mid s),\pi_\theta(a\mid s)\cdot A(s,a)\right)=Cov_{\boldsymbol{y}\sim\pi_\theta(\cdot|\boldsymbol{x})}\left(\log\pi_\theta(\boldsymbol{y}\mid\boldsymbol{x}),\pi_\theta(\boldsymbol{y}\mid x)\cdot A(\boldsymbol{y},\boldsymbol{x})\right)$$

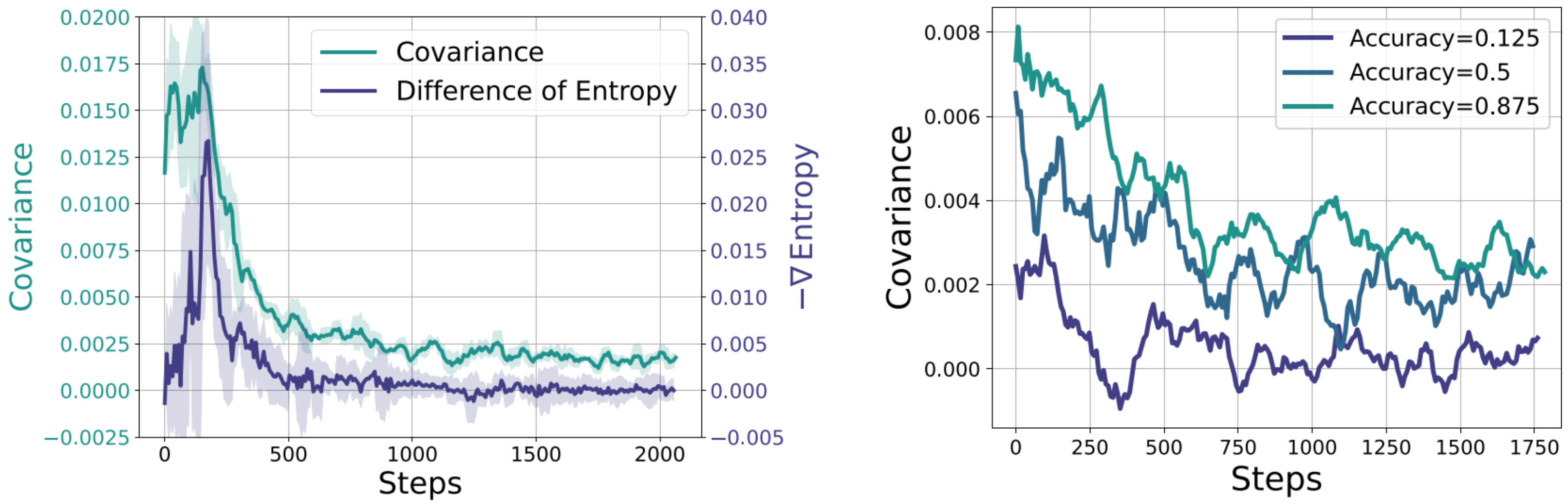


Figure 8: **Left:** The dynamics of policy entropy (step-wise entropy difference) and covariance during on-policy GRPO training. They show similar trends as expected from the theoretical results. **Right:** Different prompt groups show distinct covariance behaviors. Easier prompts with higher accuracy has higher covariances as well, while harder prompts have lower covariances.

Understanding Entropy Dynamics

TAKEAWAY

(1) For softmax policy including LLMs, the change of policy entropy is determined by the **covariance** between the log-probability and the change in logits of actions.
(2) For Policy Gradient and Natural Policy Gradient, the change in logits is proportional to the action advantage, meaning that a high covariance leads to quick decrease of policy entropy, as observed in RL for LLM reasoning.

Get Entropy Controlled

Can we directly use entropy regularization in RL?

- Entropy loss: sensitive to coefficients, no performance gain 🙁

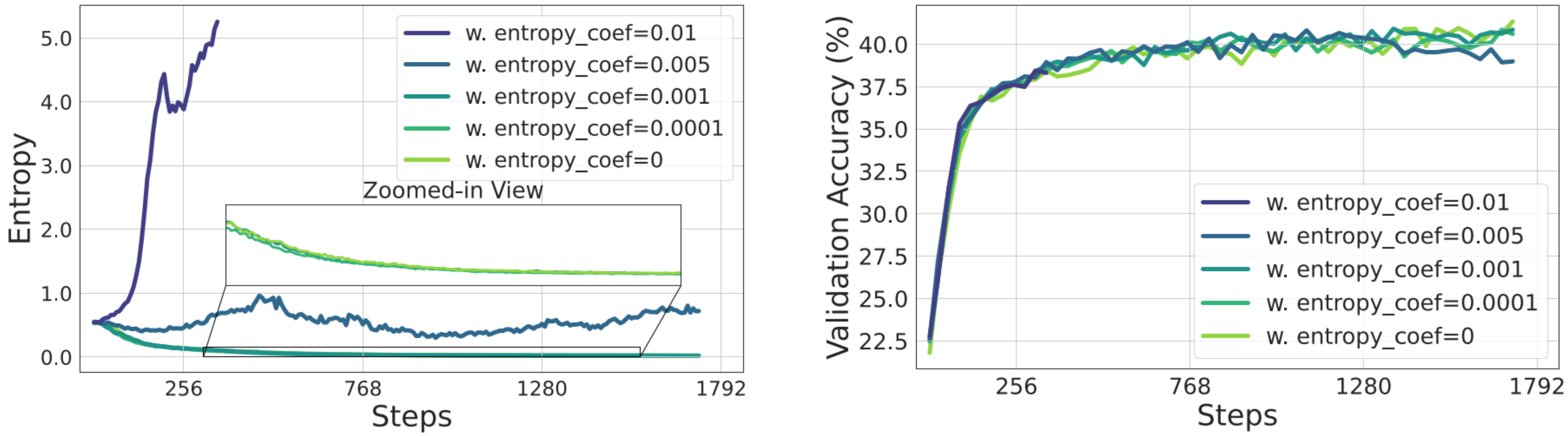


Figure 9: The policy entropy and validation accuracy of adding entropy loss where $L_{\text{ent}} = L - \alpha\mathcal{H}(\pi_\theta)$. L is the original loss and α is the coefficient of entropy loss.

Get Entropy Controlled

Can we directly use entropy regularization in RL?

- Reference KL: control entropy at the cost of performance drop 🙁

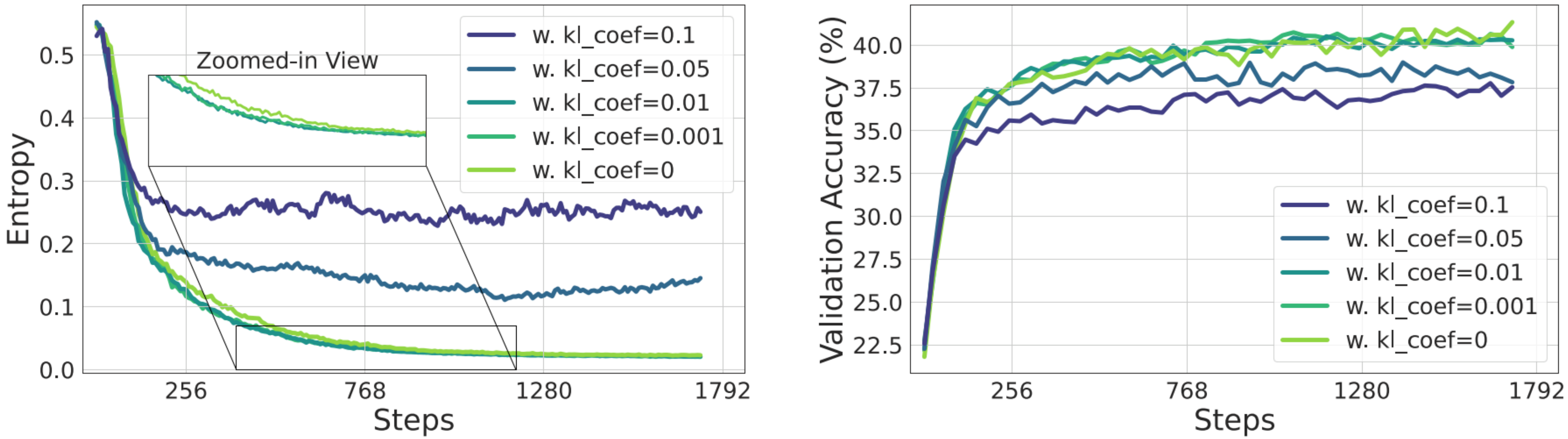


Figure 10: The policy entropy and validation accuracy of adding KL penalty between policy and reference model where $L_{\text{KL}} = L + \beta \mathbb{D}_{\text{KL}}(\pi_\theta || \pi_{\text{ref}})$. L is the original loss and β is the coefficient of KL loss.

Get Entropy Controlled

Lessons learned from entropy dynamics analysis

- All update steps have positive average covariance (100%)
- The average is *small but positive*
- There are *outliers* with high covariance (500x mean value)

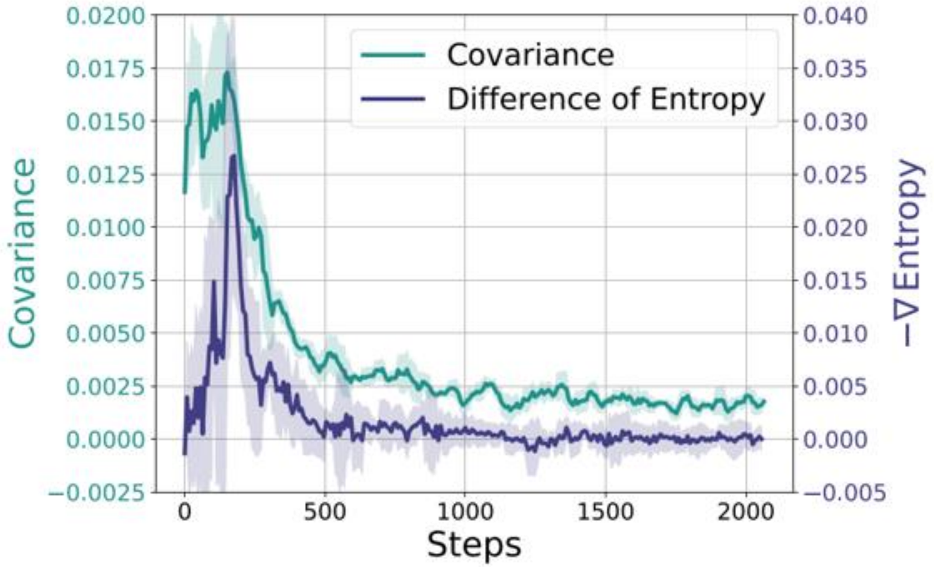


Table 1: Covariance distribution of Qwen2.5-7B in training step 1.

Group	Mean Value
Top 0.02%	5.654
Top 0.2%	3.112
Top 2%	1.385
Top 20%	0.351
Top 50%	0.152
All	0.003

Get Entropy Controlled

Guidelines for entropy control

- We only need to interfere a small portion of tokens for stability
- *Restrict the update of high-covariance tokens*

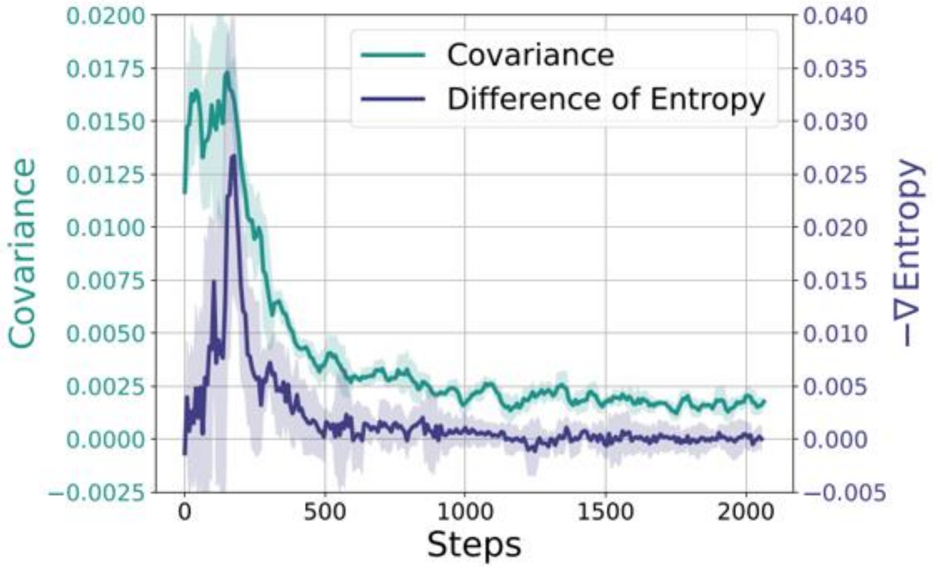


Table 1: Covariance distribution of Qwen2.5-7B in training step 1.

Group	Mean Value
Top 0.02%	5.654
Top 0.2%	3.112
Top 2%	1.385
Top 20%	0.351
Top 50%	0.152
All	0.003

Get Entropy Controlled

Guidelines for entropy control

- Two simple techniques: **`Clip-Cov`** and **`KL-Cov`**
- Strictly follow the surrogate loss in PPO

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t\Big[\min(r_t(\theta)\hat{A}_t, \operatorname{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\Big]$$

$$L^{KLPEN}(\theta) = \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}\hat{A}_t - \beta\,\mathrm{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)]\right]$$

Schulman et al., Proximal Policy Optimization Algorithms

Get Entropy Controlled

Guidelines fo

- Two simpl
- Strictly fol

```
def compute_policy_loss(old_log_prob, log_prob, advantages,
    select_ratio, method, **args):
    ratio = exp(log_prob - old_log_prob)
    pg_losses1 = -ratio * advantages
+   # calculate token wise centered cross - product
+   covs = (log_prob - log_prob.mean()) * (advantages - advantages.mean
    ())
+   select_num = int(select_ratio * len(pg_losses1))
    if method == "clip_cov":
        pg_losses2 = -clip(ratio, args["clip_range_lb"], args["
            clip_range_ub"]) * advantages
+       # randomly select index to be detached
+       clip_idx = random_select(covs[covs > args["cov_lb"] & covs <
    args["cov_ub"]], num=select_num)
+       pg_losses1[clip_idx].detach_()
+       pg_losses2[clip_idx].detach_()
        pg_loss = maximum(pg_losses1, pg_losses2).mean()
    if method == "kl_cov":
        kl_coef = args["kl_coef"]
        kl_penalty = (log_prob - old_log_prob).abs()
-       pg_losses = pg_losses1 + kl_coef * kl_penalty
+       # find out index with highest conviriance
+       select_idx = topk(covs, k=select_num, largest=True)
+       # apply KL penalty of these samples
+       pg_losses1[select_idx] += kl_coef * kl_penalty[select_idx]
        pg_loss = pg_losses1.mean()
    return pg_loss
```

Get Entropy Controlled

`Clip-Cov` and `KL-Cov`

- They indeed get entropy controlled

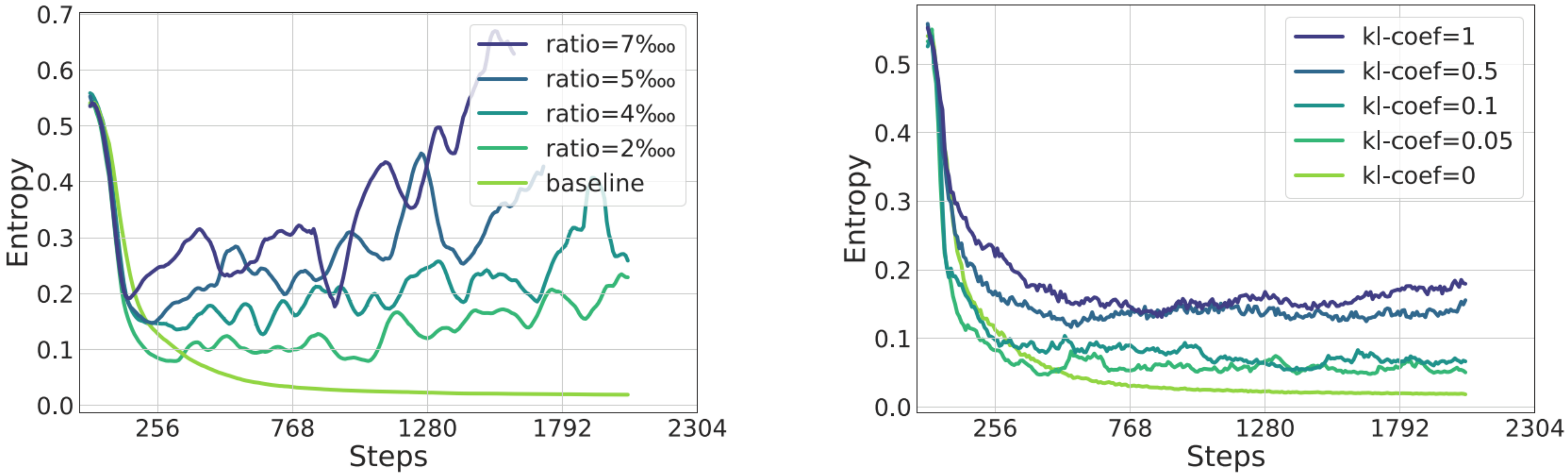


Figure 12: Differences in entropy dynamics of Qwen2.5-7B under varying KL coefficients and clip ratios, evaluated in `KL-Cov` and `Clip-Cov` settings, respectively.

Get Entropy Controlled

`Clip-Cov` and `KL-Cov`

- Get higher entropy and better performance

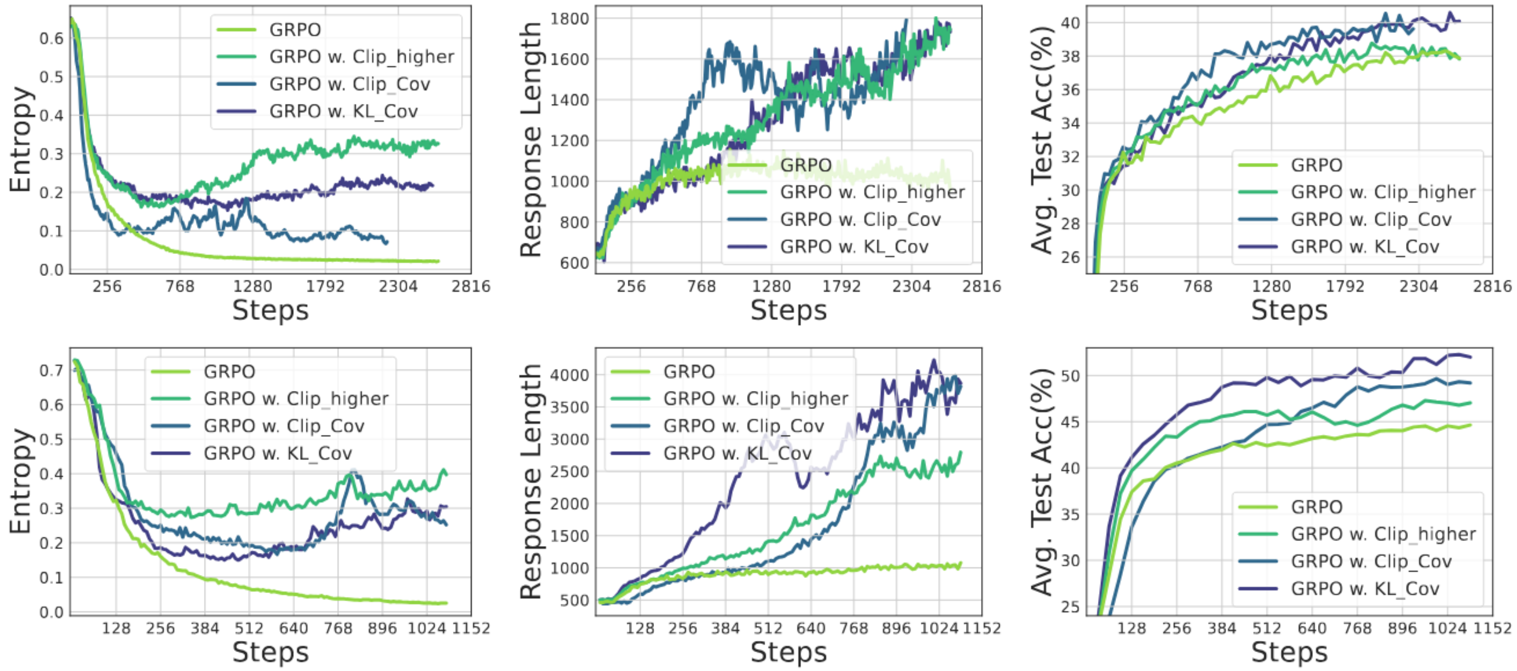


Figure 11: Training Qwen2.5-7B (Up) / Qwen2.5-32B (Down) with GRPO with/without our methods. **Left:** Entropy dynamics. Our methods uplift policy entropy from collapse, enabling sustained exploration. **Middle:** Our method also incentivizes longer responses compared with vanilla GRPO. **Right:** The policy model consistently outperforms baseline on testsets, avoiding performance plateaus.

Get Entropy Controlled

`Clip-Cov` and `KL-Cov`

- Get higher entropy and better performance

Table 2: Detailed results of GRPO, GRPO with clip-higher technique and our methods. For AIME and AMC, the results are avg.@32. **Bold** denotes the best results.

Method	AIME24	AIME25	AMC	MATH-500	OMNI-MATH	OlympiadBench	Minerva	Avg.
Qwen2.5-7B								
GRPO	21.2	9.6	58.7	78.8	27.9	40.7	36.7	38.6
w. Clip-higher	18.1	11.5	56.6	79.2	29.8	43.3	40.4	38.8
w. CLIP-Cov	22.1	**15.8**	58.2	80.4	**30.5**	**44.1**	**41.1**	40.4
w. KL-Cov	**22.6**	12.9	**61.4**	**80.8**	29.1	42.6	38.2	**40.6**
Qwen2.5-32B								
GRPO	21.8	16.2	69.7	84.2	35.2	43.6	45.5	45.8
w. Clip-higher	35.6	22.3	69.5	77.2	35.1	42.5	43.0	47.2
w. CLIP-Cov	32.3	22.7	67.2	**87.0**	**42.0**	**57.2**	46.0	50.3
w. KL-Cov	**36.8**	**30.8**	**74.5**	84.6	39.1	49.0	**46.3**	**52.2**

Get Entropy Controlled

TAKEAWAY

We can control policy entropy by **restricting the update of tokens with high covariances**, e.g., clipping (`Clip-Cov`) or applying KL penalty (`KL-Cov`). These simple techniques prevent policy from entropy collapse thus promote exploration.

Closing Thoughts

- LLMs are general-purpose, strong priors as the policy in RL
- As expected, we see improvements in many fields
- However, most RL is just **reinforcing the self confidence** of LLMs, make it a more stable but less exploratory policy
- Stronger model with narrower distribution
- *Is it a blessing or a curse?*

Paper

Code

Thanks!

Ganqu Cui, Yuchen Zhang*, Jiacheng Chen*, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, Zhiyuan Liu, Hao Peng, Lei Bai, Wanli Ouyang, Yu Cheng, Bowen Zhou, Ning Ding*

2025.07.01