JustRL: Scaling a 1.5B LLM with a Simple RL Recipe

Bingxiang He | THUNLP | Advisor: Prof. Zhiyuan Liu

Homepage: https://hbx-hbx.github.io/

2026.02.03

JustRL: Scaling a 1.5B LLM with a Simple RL Recipe

Bingxiang He[1], **Zekai Qu**[1] , **Zeyuan Liu**[1] , **Yinghao Chen**[1], **Yuxin Zuo**[1], **Cheng Qian**[2], **Kaiyan Zhang**[1], **Weize Chen**[1], **Chaojun Xiao**[1], **Ganqu Cui**[3], **Ning Ding**[1†], **Zhiyuan Liu**[1†]

[1]Tsinghua University [2]University of Illinois Urbana-Champaign [3]Shanghai AI Lab

[†]**Corresponding Authors.** hebx24@mails.tsinghua.edu.cn

https://huggingface.co/collections/hbx/justrl

https://github.com/thunlp/JustRL

"Perfection is achieved, not when there is nothing more to add, but when there is nothing left to take away."

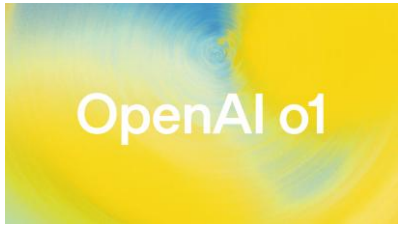
——— Antoine de Saint-Exupéry, Airman's Odyssey

Outline

- Background
- JustRL Recipe
- Experiments
- Discussion

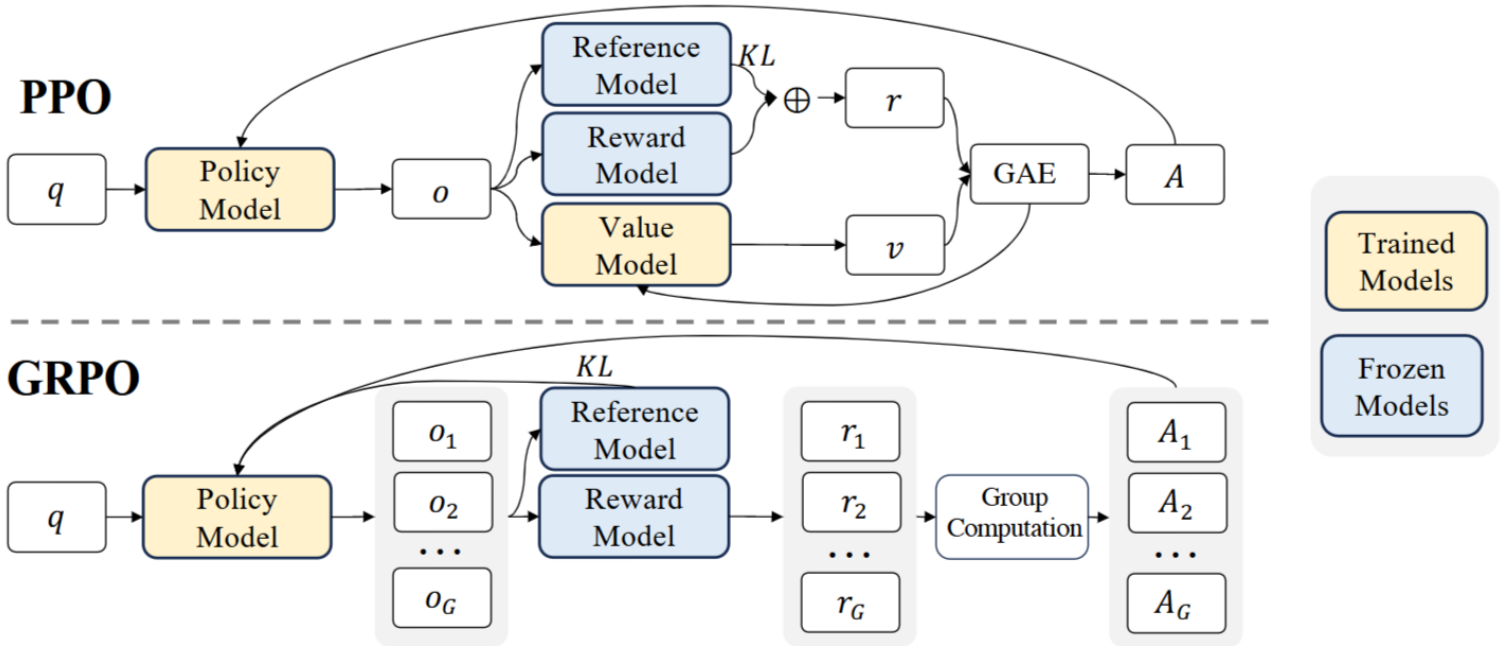
Background

- **2025 is a year of reasoning models trained via large scale RL**
  - Extended thinking through long CoT
  - SoTA performance on challenging reasoning benchmarks



Background

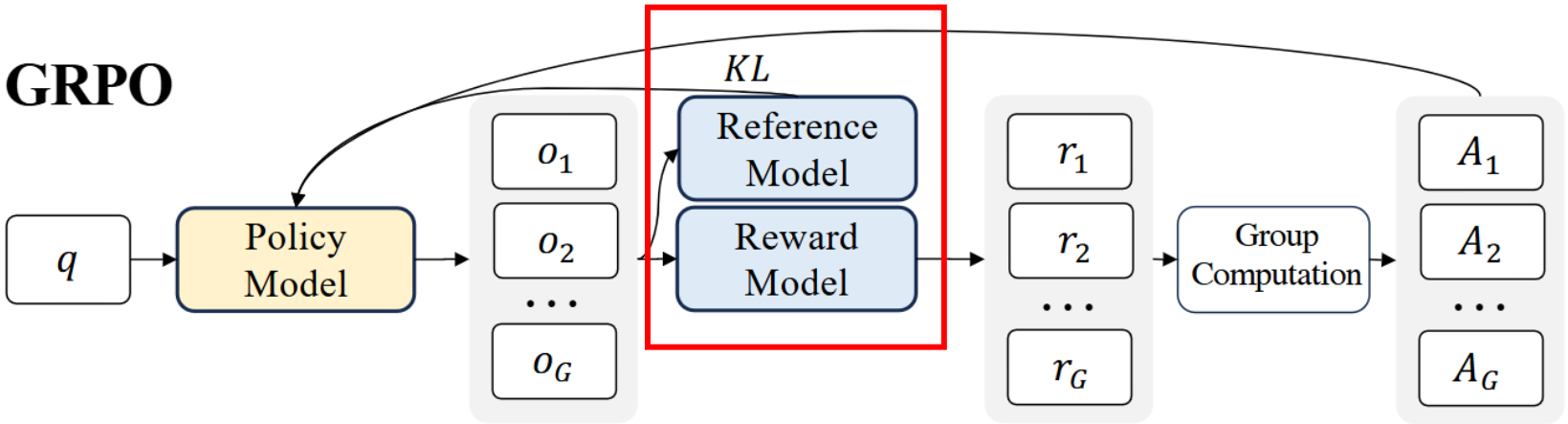
- **How expensive is RL training today?**



From PPO to GRPO, we have eliminated the value model

Background

➢ **How expensive is RL training today?**

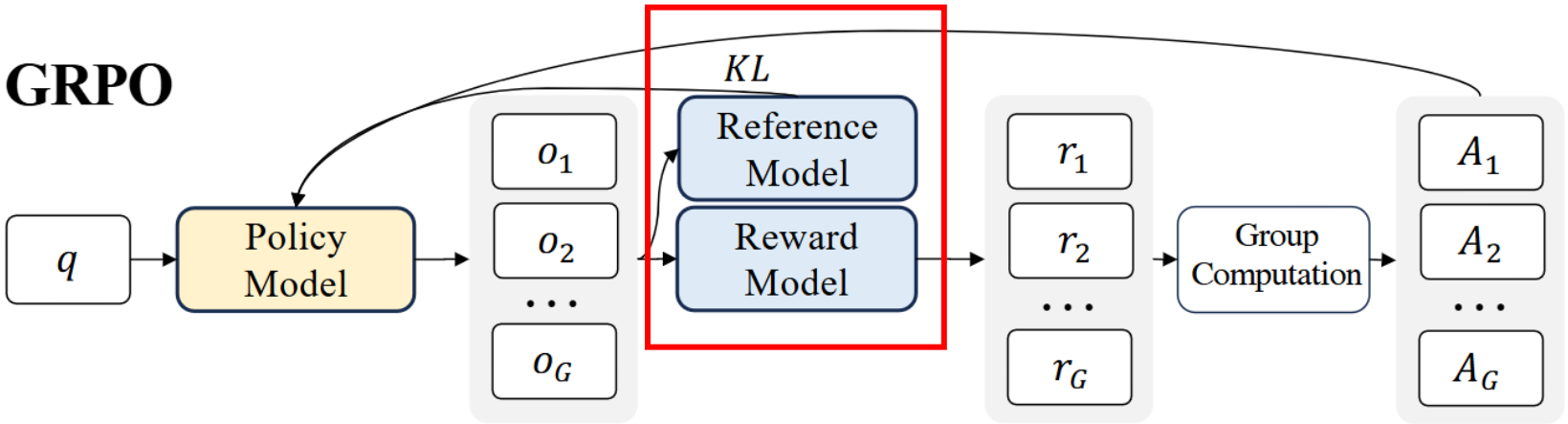


For GRPO, we can further eliminate

- Reference Model: w/o KL regularization
- Reward Model: From RLHF to RLVR (Verifiable Rewards)

Background

- **How expensive is RL training today?**

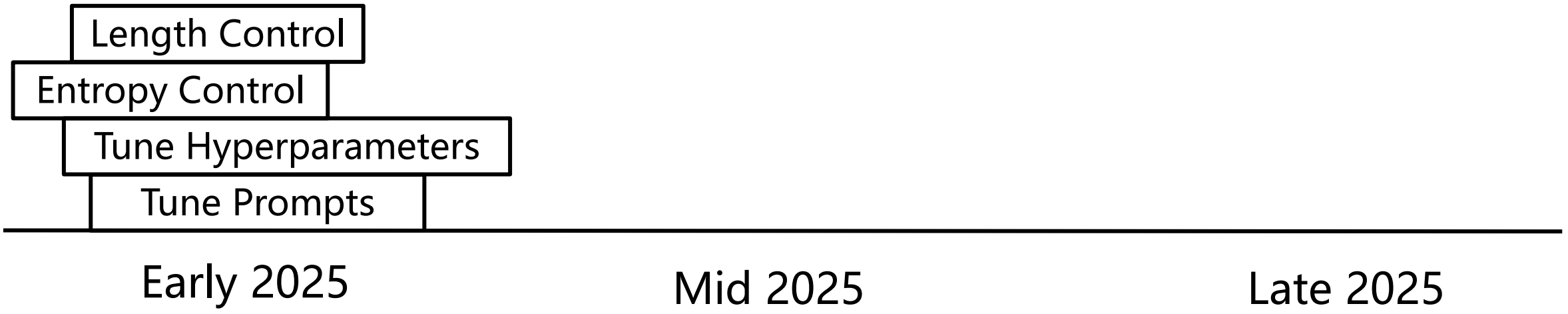


Even with this minimal setup, the cost is still high

- 32× A800-80GB GPUs
- Batch size 64, n=8 rollouts, 16k max context
- ~5 minutes per training step

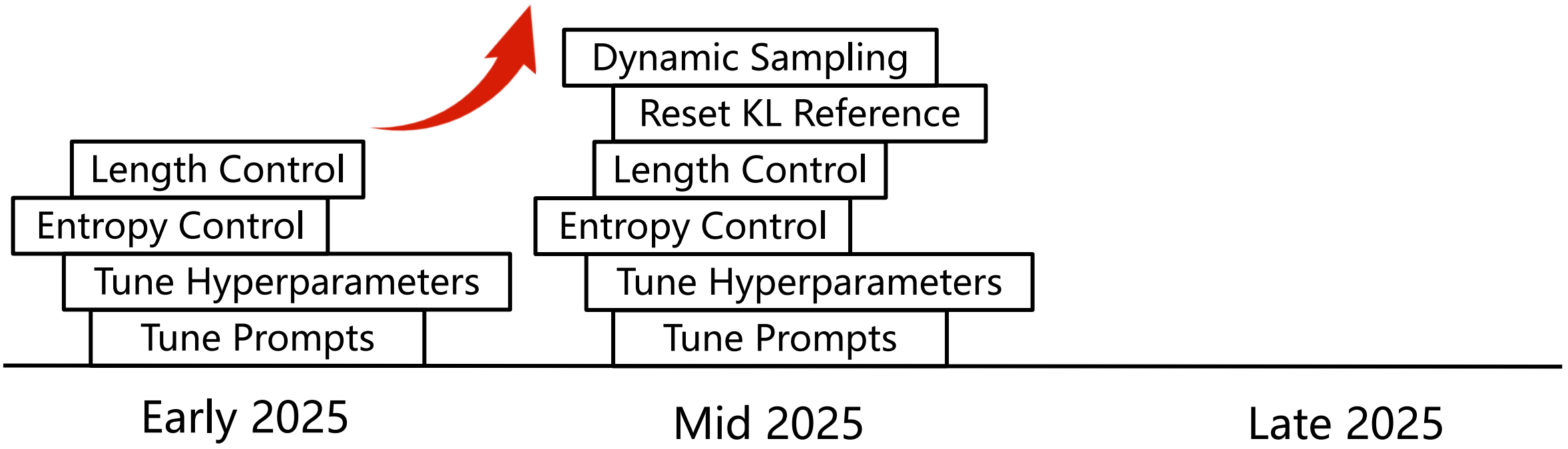
Background

- **How complex is RL training today?**



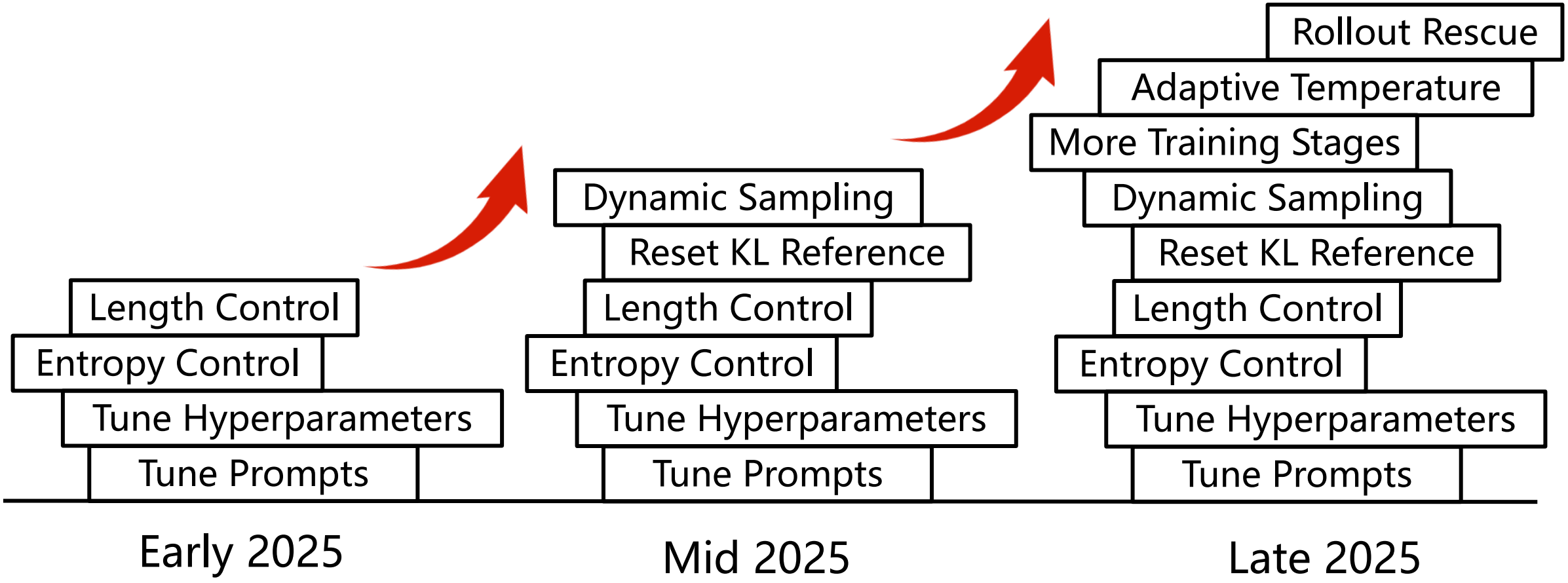
Background

- **How complex is RL training today?**



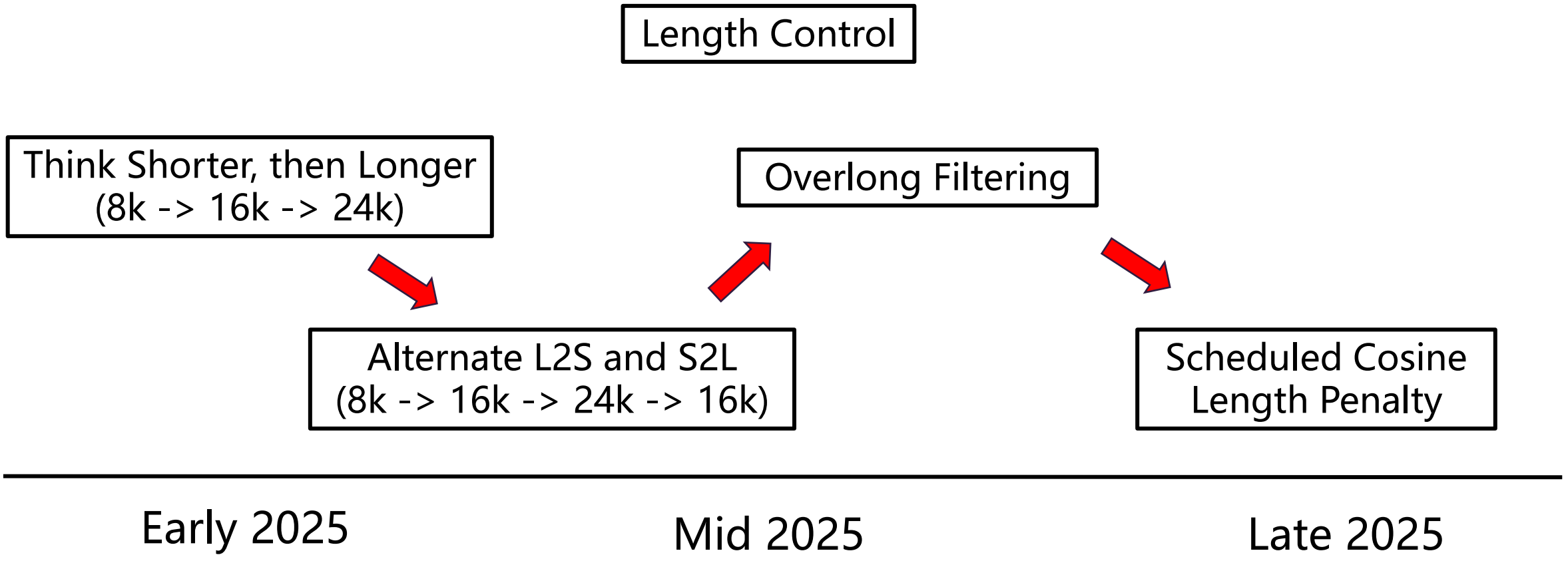
Background

- **How complex is RL training today?**



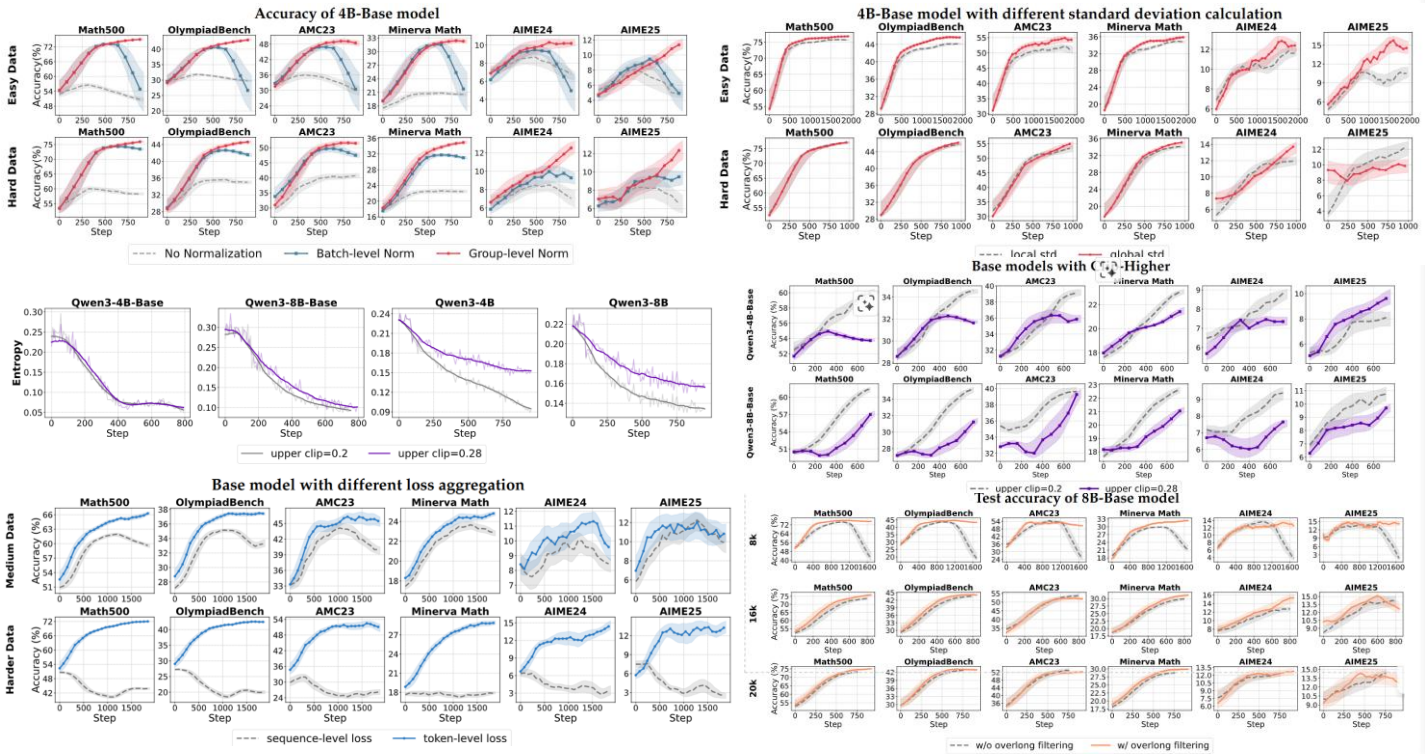
Background

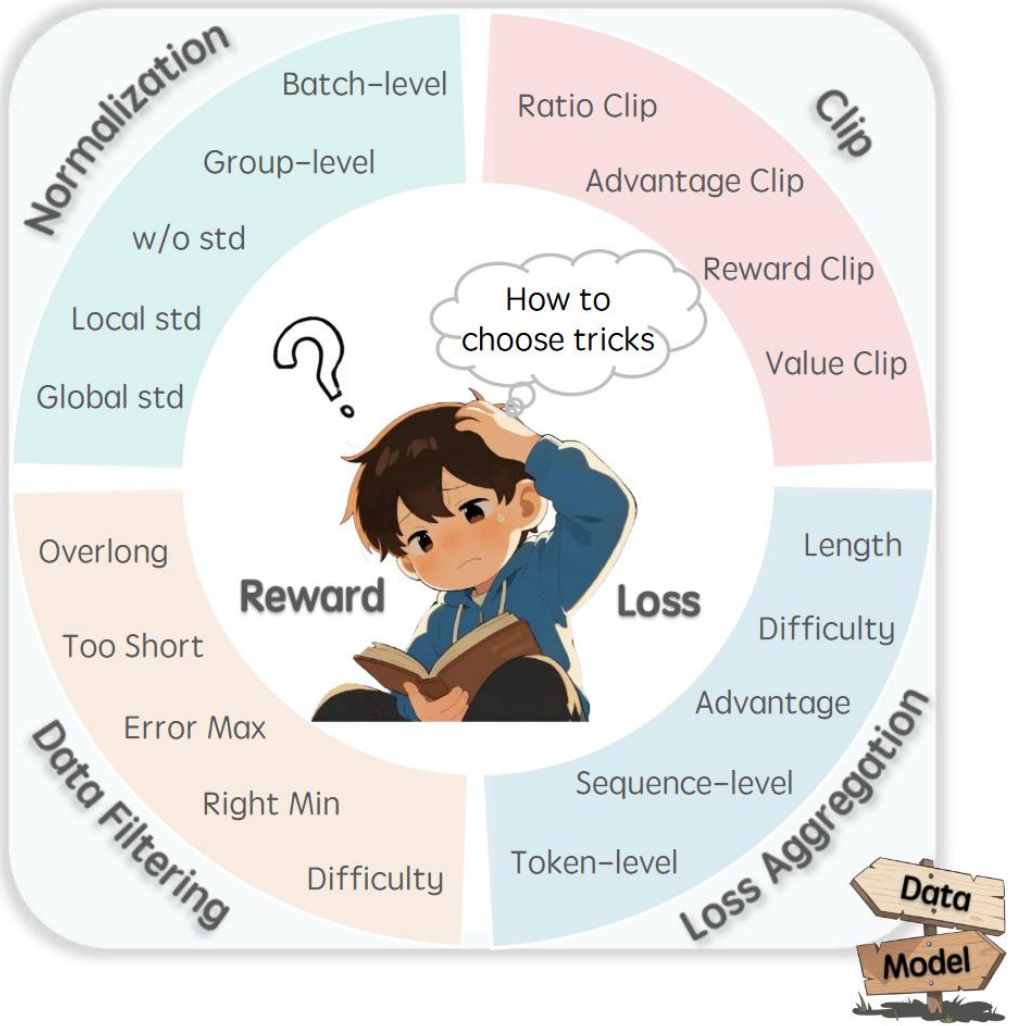
- **How complex is RL training today?**



Background

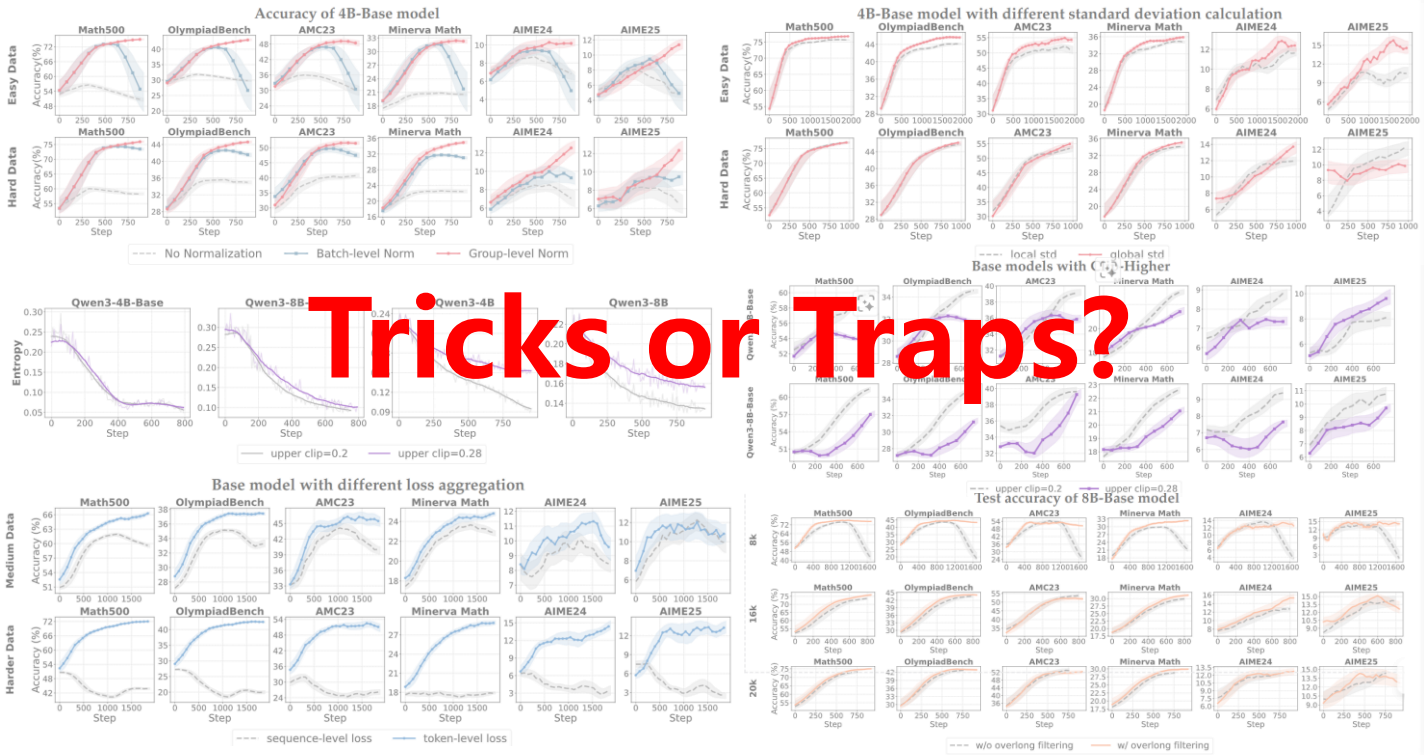
- **How complex is RL training today?**



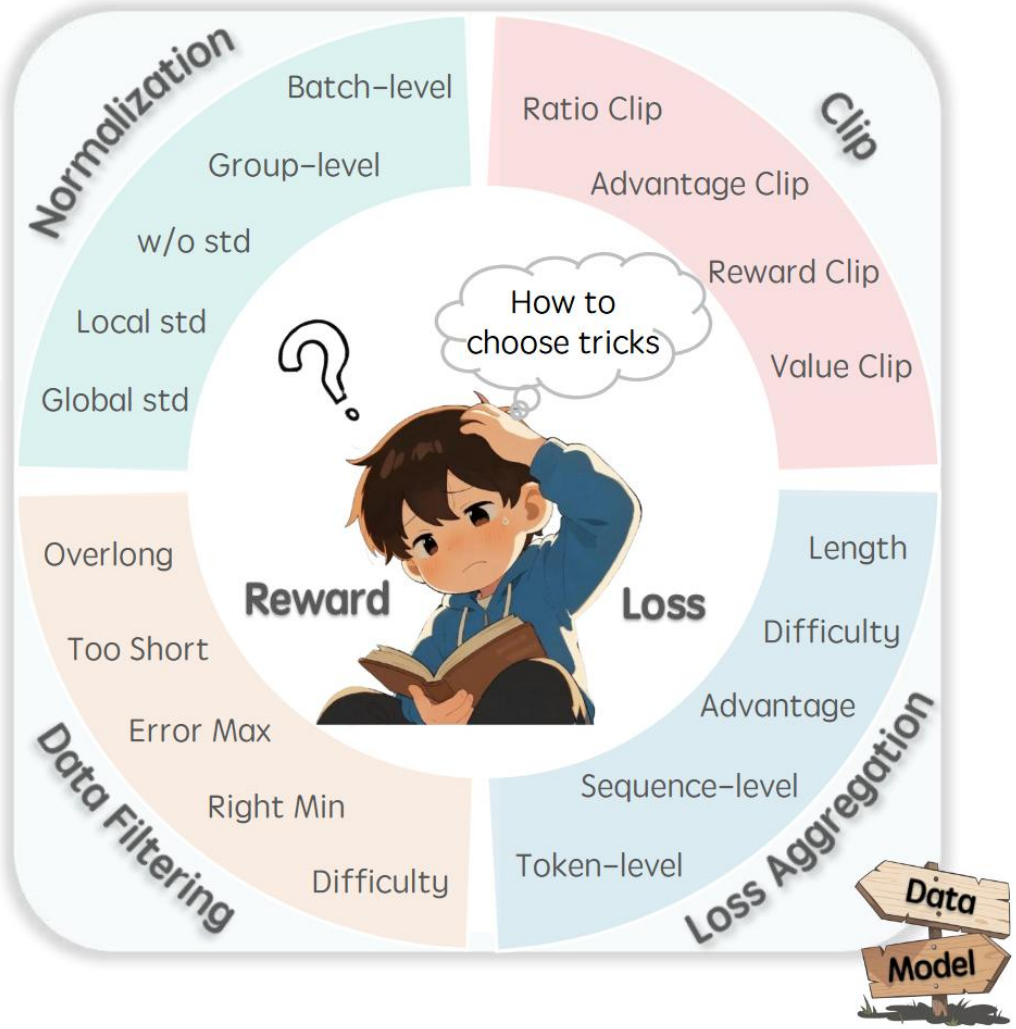


Background

- **How complex is RL training today?**

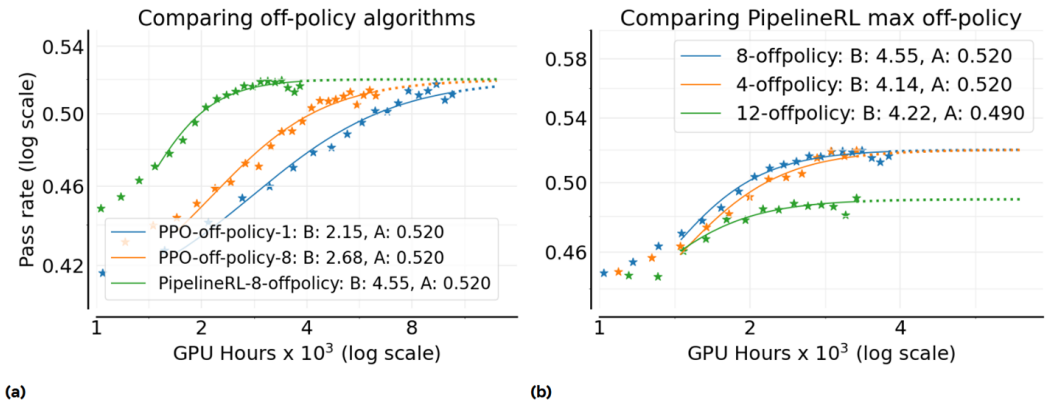


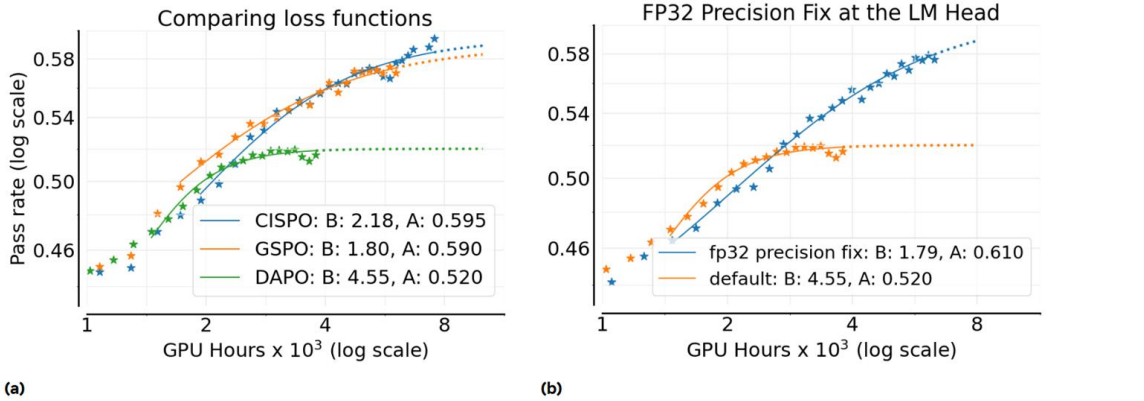
Tricks or Traps?

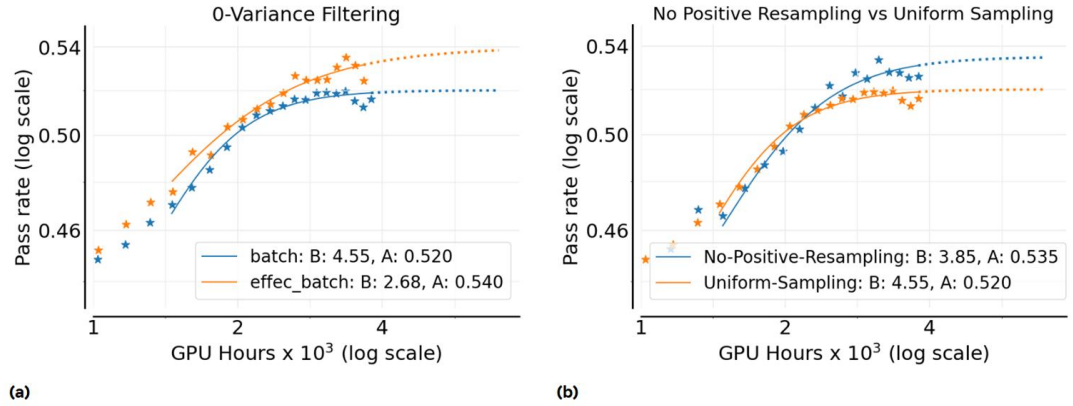


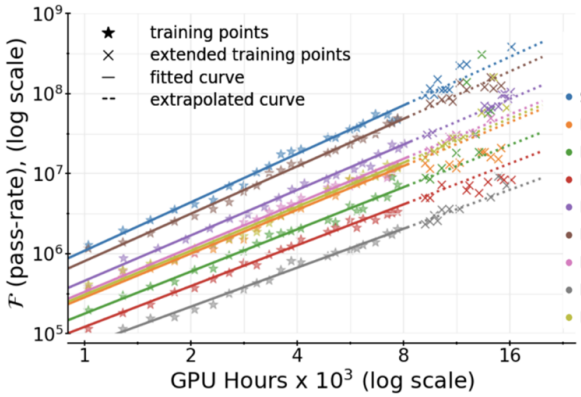
Background

➢ How complex is RL training today?





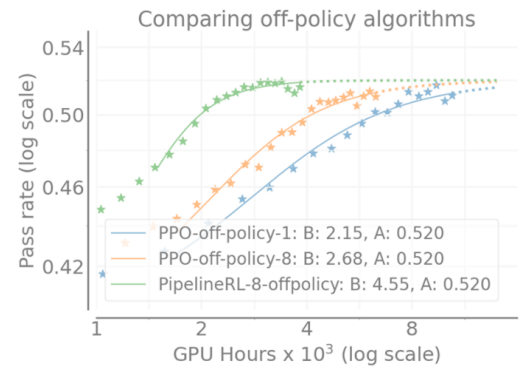


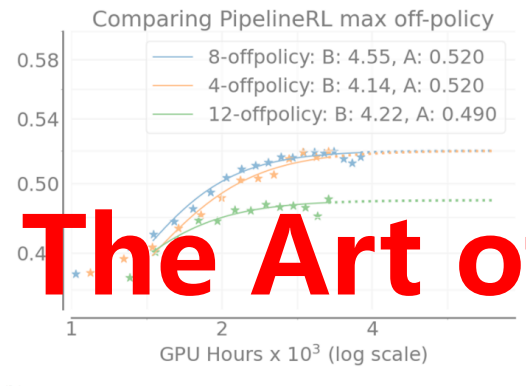


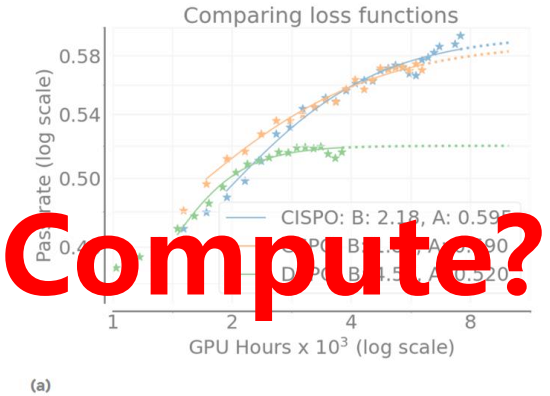
Method	fitted B w/ fixed A = 0.605	original fitted parameters C_{mid} / B / A
ScaleRL	2.01	2542 / 1.92 / 0.610
LOO-prompt-lvl-adv-norm	1.82	2463 / 2.09 / 0.590
LOO-length-penalty	1.75	2660 / 1.68 / 0.610
LOO-batch	1.69	2503 / 1.84 / 0.595
LOO-PPO-off-policy-8	1.89	3409 / 1.89 / 0.605
LOO-no-fp32-precision-fix	1.97	2857 / 1.88 / 0.610
LOO-sample-avg	1.84	2621 / 1.93 / 0.600
LOO-uniform-sampling	1.62	2621 / 1.62 / 0.605
LOO-dapo	1.83	3093 / 1.77 / 0.610

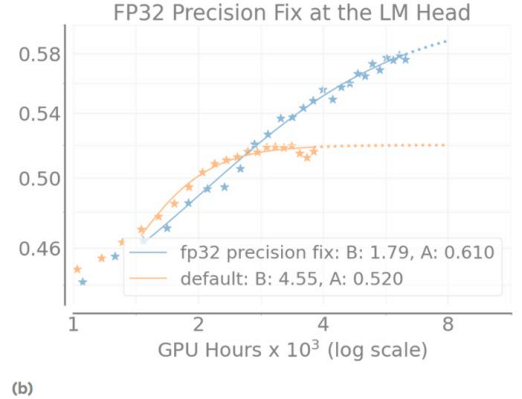
Background

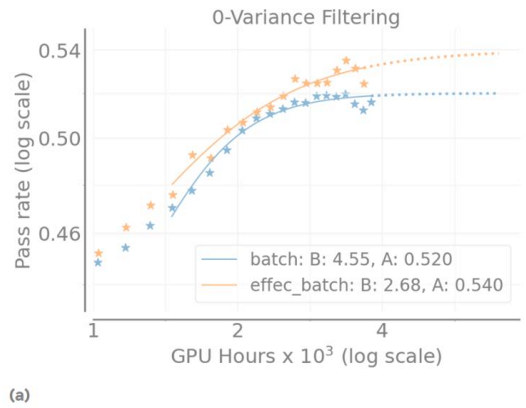
- **How complex is RL training today?**

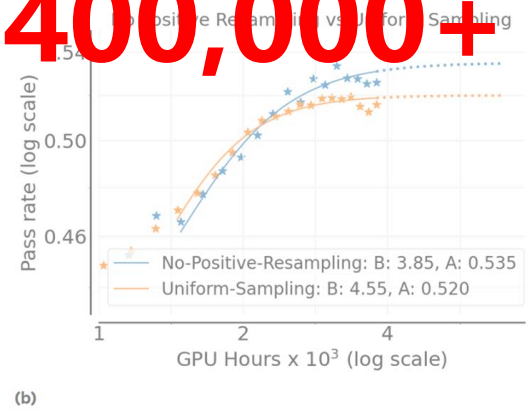


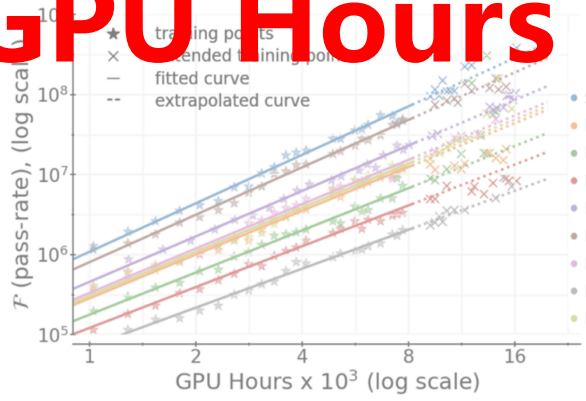












Method	fitted B w/ fixed A = 0.605	original fitted parameters C_{mid} / B / A
ScaleRL	2.01	2542 / 1.92 / 0.610
LOO-prompt-lvl-adv-norm	1.82	2463 / 2.09 / 0.590
LOO-length-penalty	1.75	2660 / 1.68 / 0.610
LOO-batch	1.69	2503 / 1.84 / 0.595
LOO-PPO-off-policy-8	1.89	3409 / 1.89 / 0.605
LOO-no-fp32-precision-fix	1.97	2857 / 1.88 / 0.610
LOO-sample-avg	1.84	2621 / 1.93 / 0.600
LOO-uniform-sampling	1.62	2621 / 1.62 / 0.605
LOO-dapo	1.83	3093 / 1.77 / 0.610

The Art of Compute?

400,000+ GPU Hours

Background

➢ **How complex is RL training today?**

Different backbones

- DeepSeek-R1-Distill-Qwen-1.5B
- OpenMath-Nemotron-1.5B
- Qwen3-1.7B

Model	EC	THP	TTP	RKL	LC	AT	RR	DS	ST	Date
STILL-3-1.5B	✗	✓	✓	✓	✗	✗	✗	✗	✗	Jan '25
DeepScaleR-1.5B	✓	✗	✗	✗	✓	✗	✗	✗	✓	Feb '25
FastCuRL-1.5B	✗	✓	✗	✗	✓	✗	✗	✗	✓	Mar '25
ProRL-V1	✓	✓	✗	✓	✓	✗	✗	✓	✓	May '25
e3-1.7B	✓	✓	✗	✗	✓	✗	✗	✓	✓	Jun '25
POLARIS-1.7B	✓	✓	✗	✗	✓	✓	✓	✓	✓	Jul '25
ProRL-V2	✓	✓	✗	✓	✓	✗	✗	✓	✓	Aug '25
QuestA-Nemotron	✗	✗	✗	✗	✗	✗	✗	✓	✓	Sep '25
BroRL	✓	✓	✗	✓	✓	✗	✗	✓	✓	Oct '25
JustRL-DeepSeek	✓	✗	✗	✗	✗	✗	✗	✗	✗	Nov '25
JustRL-Nemotron	✓	✗	✗	✗	✗	✗	✗	✗	✗	Nov '25

Table 1 | Comparison of RL techniques used in recent small language models for mathematical reasoning. Model names are colored by backbone: DeepSeek-R1-Distill-Qwen-1.5B, Qwen3-1.7B, OpenMath-Nemotron-1.5B. We use the following abbreviations for RL techniques: EC=Entropy Control, THP=Tune Hyperparameters, TTP=Tune Training Prompt, RKL=Reset KL Reference, LC=Length Control, AT=Adaptive Temperature, RR=Rollout Rescue, DS=Dynamic Sampling, ST=Split Training Stages. Our models (JustRL-DeepSeek and JustRL-Nemotron) use only entropy control, achieving competitive performance with minimal complexity.

Background

➢ **How complex is RL training today?**

Rollout Rescue | Adaptive Temperature | More Training Stages | ……

…… | Dynamic Sampling | Reset KL Reference | Length Control

Entropy Control | Tune Hyperparameters | Tune Prompts | ……

RL Researcher's Dilemma

Background

- **How complex is RL training today?**

Rollout Rescue | Adaptive Temperature | More Training Stages | ……

…… | Dynamic Sampling | Reset KL Reference | Length Control

Entropy Control | Tune Hyperparameters | Tune Prompts | ……

Apply them all / Test them one by one

Background

- **How complex is RL training today?**

KL Drift

Stage 3 Collapse

Prompt Sensitivity

Hyperparam Search

Gradient Explosion

Reward Hacking

Loss Spike

……

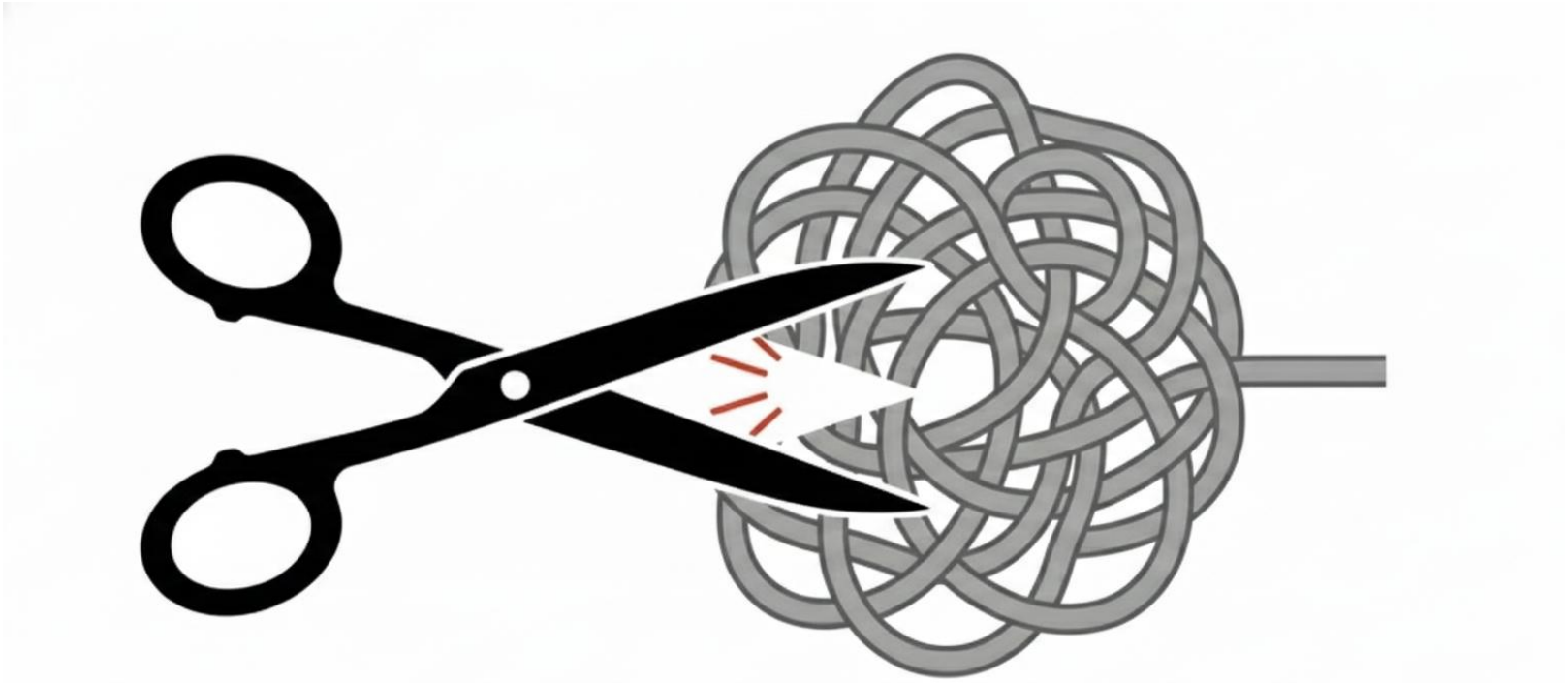
10+ tricks to validate
× **2** weeks per ablation (**32** gpus)

Background

- **The real problem:**
  - Tricks did work in some scenarios, but not always
  - Different backbones, datasets, hyperparameters → different results
  - Each new trick = +1 tunable parameter = +instability

- **Our motivation:**
  - Maybe we're adding complexity to solve problems created by other complexity
  - What if stable, competitive training can be achieved with a simpler approach?
  - Can we build a strong baseline instead?

Background



What happens if we stop adding, and start subtracting?

Outline

- Background
- **JustRL Recipe**
- Experiments
- Discussion

JustRL Recipe

- **What we keep simple?**

~~Multi-Stage Training~~	Single-Stage Training w Clip Higher
~~Dynamic Sampling~~	Fixed Hyperparam
~~Prompt Engineering~~	w/o Dynamic Sampling
~~Adaptive Temperature~~	Basic Prompting
~~Reset KL Reference~~	w/o Length Penalty
~~Length Control~~	w/o KL regularization
……	……

JustRL Recipe

➢ Training setting

- Framework: veRL
- Dataset: DAPO-math-17k (repeated 100 times)
- Backbones: DeepSeek-R1-Distill-Qwen-1.5B and OpenMath-Nemotron-1.5B
- 32 A800-80GB GPUs for ~15 days
- Fixed hyperparams

Hyperparameter	Value
Advantage Estimator	GRPO
Use KL Loss	No
Use Entropy Regularization	No
Train Batch Size	256
Max Prompt Length	1k
Max Response Length	15k
PPO Mini Batch Size	64
PPO Micro Batch Size/GPU	1
Clip Ratio Range	[0.8, 1.28]
Learning Rate	1e-6 (constant)
Temperature	1.0
Rollout N	8
Reward Function	DAPO [Yu et al., 2025]

JustRL Recipe

➢ Evaluation setting

- **9** Challenging Math Benchmarks (Pass@1):
  - AIME 2024 (avg@32), AIME 2025 (avg@32), AMC 2023 (avg@32)
  - MATH-500 (avg@4), Minerva Math (avg@4), OlympiadBench (avg@4)
  - HMMT Feb 2025 (avg@32), CMIMC 2025 (avg@32), and BRUMO 2025 (avg@32)
- Configs:
  - Temperature 0.7, Top-p 0.9, Max Tokens 32k
  - Rule-based verifier adapted from POLARIS
  - Augmented with model-based verifier CompassVerifier-3B

https://github.com/thunlp/JustRL

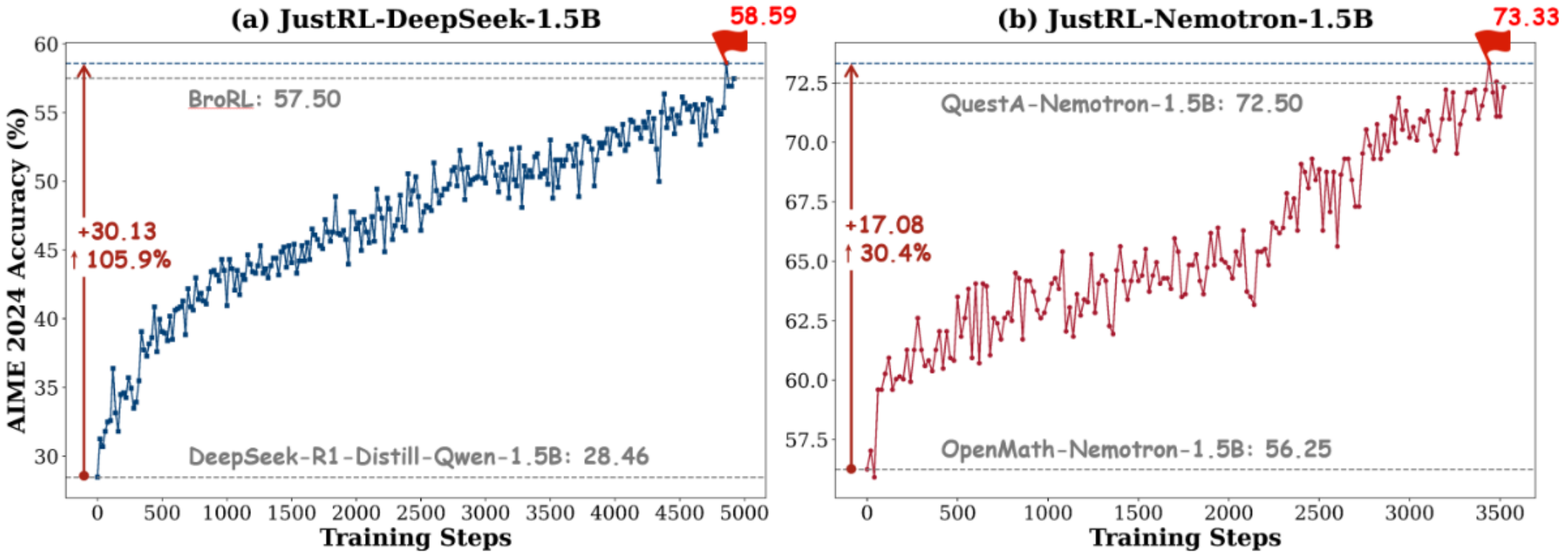
Outline

- Background
- JustRL Recipe
- **Experiments**
- Discussion

Experiments

➢ Overview

- From Dpsk-Distill, JustRL-DeepSeek-1.5B achieves from 28% to 58% over 4,000 steps
- From Nemotron, JustRL-Nemotron-1.5B achieves final 70+ on AIME24



Experiments

➢ Scaling a weaker base: JustRL-DeepSeek-1.5B (4,380 steps)

- Achieves 54.87% average across benchmarks, outperforming ProRL-V2′ s 53.08%
- Despite ProRL-V2′ s nine-stage training pipeline with dynamic hyperparameters and more sophisticated techniques
- Lead on six of nine benchmarks

Model	AIME24	AIME25	AMC23	MATH	Minerva	Olympiad	HMMT	BRUMO	CMIMC	Avg
Backbone	29.90	22.40	63.82	84.90	34.65	45.95	13.44	30.94	12.89	37.65
DeepScaleR-1.5B	40.21	28.65	73.83	89.30	39.34	52.79	18.96	40.00	21.00	44.88
ProRL-V2	51.87	35.73	88.75	92.00	49.03	67.84	19.38	47.29	**25.86**	53.08
BroRL*	**57.50**	36.88	–	**92.14**	49.08	61.54	–	–	–	–
JustRL-DeepSeek	52.60	**38.75**	**91.02**	91.65	**51.47**	**67.99**	**21.98**	**52.71**	25.63	**54.87**

Table 3 | Results on DeepSeek-R1-Distill-Qwen-1.5B backbone. All scores except MATH-500, Minerva, and OlympiadBench use @32 sampling; those three use @4. *BroRL results are officially reported but models not released; some benchmarks unavailable.

Experiments

➢ Scaling a weaker base: JustRL-DeepSeek-1.5B (4,380 steps)

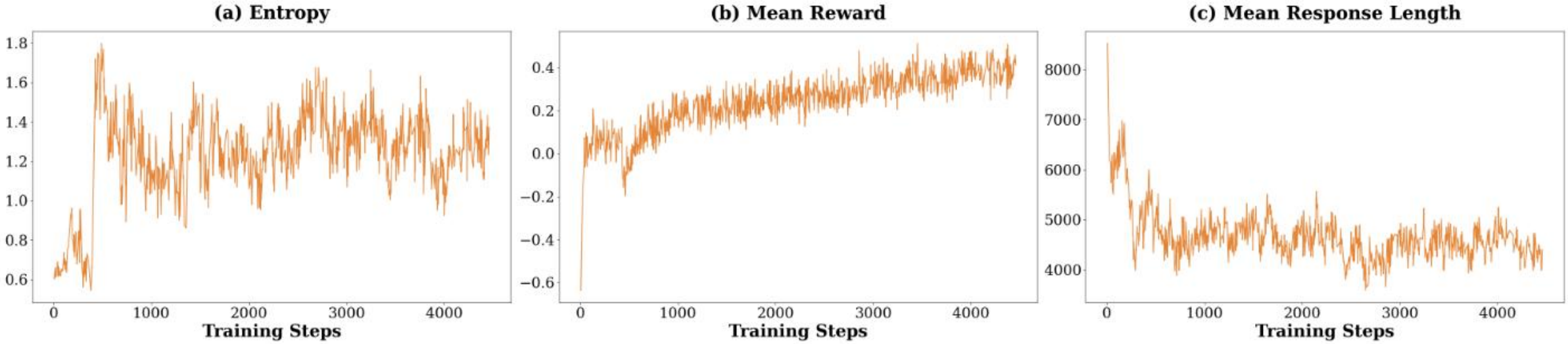
- We match half of ProRL-V2' s compute budget while using a single-stage recipe with fixed hyperparameters
- BroRL requires 4.9× more compute by increasing rollouts to 512 per example

Model	Dynamic Sampling*	Training Steps	Train Batch Size	Rollout N	Max Context Length	Token Budget (approx.)
DeepScaleR-1.5B	✗	1,750	128	8	8k→16k→24k	2.2×10^{6}k
ProRL-V1	✔	2,450	256	16→32→16	8k→16k	2.1×10^{8}k
ProRL-V2	✔	+1,000	256	16→32→16	8k→16k→8k	2.8×10^{8}k
BroRL	✔	+191	128	512	16k	6.8×10^{8}k
JustRL-DeepSeek	✗	4,380	256	8	16k	1.4×10^{8}k

Table 4 | Computational cost comparison for DeepSeek-R1-Distill-Qwen-1.5B based models. *Dynamic sampling with estimated 50% filter ratio following POLARIS [An et al., 2025]. ProRL-V2 continues from ProRL-V1 (+1,000 steps), and BroRL continues from ProRL-V2 (+191 steps).

Experiments

- **Scaling a weaker base: JustRL-DeepSeek-1.5B (4,380 steps)**
  - Entropy oscillating between 1.0 and 1.6 at later training steps, simple clip higher technique is well-performed for large-scale RL
  - Reward climbing from around -0.6 to +0.4 over training
  - Response length naturally compresses to 4,000-5,000 tokens without any explicit length penalty, in line with other works like DLER (Doing Length pEnalty Right)



Experiments

➢ Scaling a stronger base: JustRL-Nemotron-1.5B (3,440 steps)

- Achieves 64.32% average, slightly outperforming QuestA’ s 63.81%
- QuestA augments questions with partial CoT solutions as hints, requiring full reasoning trajectories generated by larger models for curriculum construction
- Lead on five of nine benchmarks

Model	AIME24	AIME25	AMC23	MATH	Minerva	Olympiad	HMMT	BRUMO	CMIMC	Avg
Backbone	58.75	48.44	90.55	92.40	26.93	71.70	30.10	61.67	30.08	56.74
QuestA	**71.56**	62.08	93.44	92.95	**32.08**	72.28	**40.94**	**67.50**	41.48	63.81
JustRL-Nemotron	69.69	**62.92**	**96.02**	**94.15**	30.24	**76.59**	40.63	66.88	**41.72**	**64.32**

Table 5 | Results on OpenMath-Nemotron-1.5B backbone. All scores except MATH-500, Minerva, and OlympiadBench use @32 sampling; those three use @4.

Experiments

➢ Scaling a stronger base: JustRL-Nemotron-1.5B (3,440 steps)

- We use 2× less compute while achieving slightly better average performance without designing a complex curriculum as used in QuestA

Model	Dynamic Sampling*	Training Steps	Train Batch Size	Rollout N	Max Context Length	Token Budget (approx.)
QuestA	✔	2,000	128	16	32k	2.6×10^8k
JustRL-Nemotron	✗	3,440	256	8	16k	1.1×10^8k

Table 6 | Computational cost comparison for OpenMath-Nemotron-1.5B based models. *Dynamic sampling with estimated 50% filter ratio. Despite more training steps, JustRL-Nemotron uses 2.4× less compute.

Experiments

➢ Contrast with typical RL

- ProRL: observe length drift -> introduce scheduled length penalties
- BroRL: hitting plateaus -> increase rollouts to hundreds
- ……

Experiments

➢ Contrast with typical RL

- ProRL: observe length drift -> introduce scheduled length penalties
- BroRL: hitting plateaus -> increase rollouts to hundreds
- ……

➢ What this suggests

- Simpler is always better? ✗
- Simpler might actually be good ✓
- JustRL simply don’ t require the interventions that have become standard practice

Experiments

➢ Contrast with typical RL

- ProRL: observe length drift -> introduce scheduled length penalties
- BroRL: hitting plateaus -> increase rollouts to hundreds
- ……

➢ What this suggests

- Simpler is always better? ✗
- Simpler might actually be good ✓
- JustRL simply don’ t require the interventions that have become standard practice

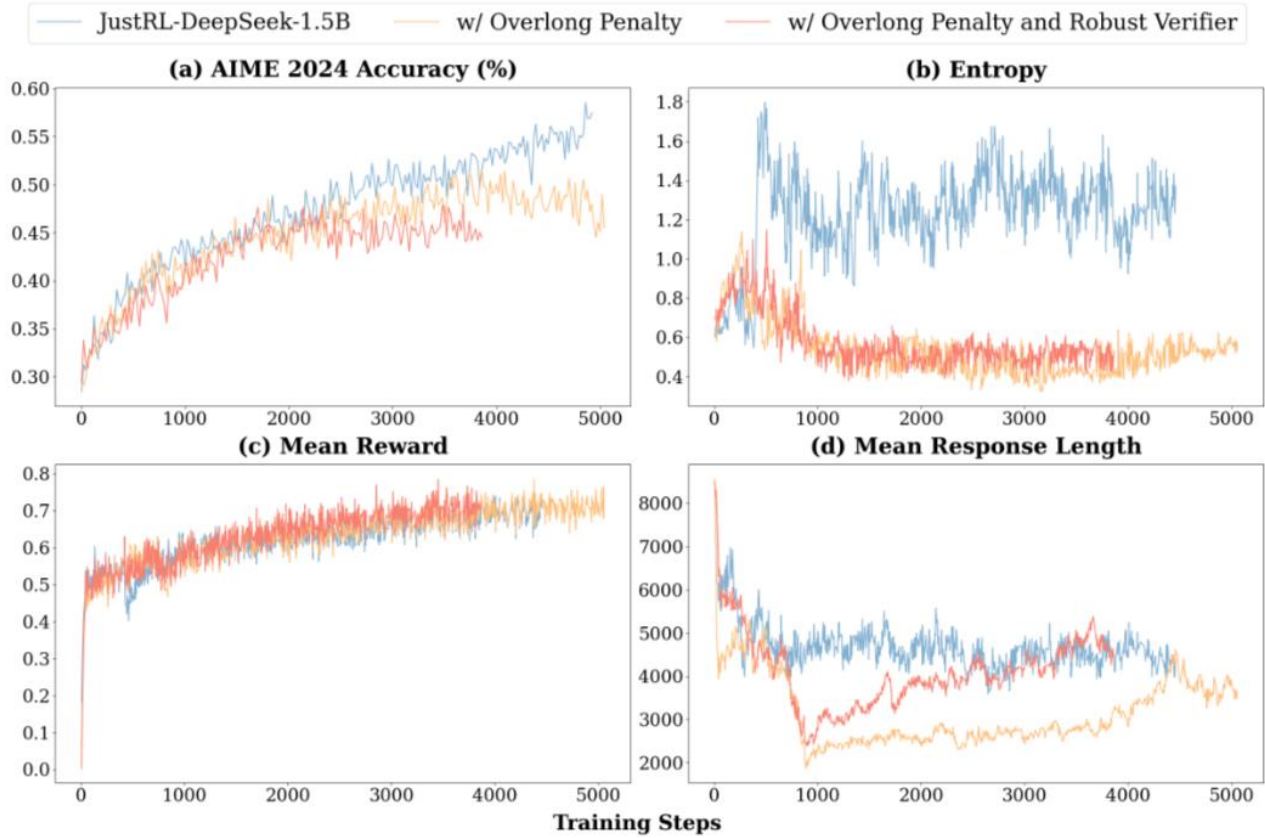
➢ So why? What contributes to this?

- We did want to know, so have a try in two following ablations

Experiments

➢ When standard tricks actually hurt

- **w/ Overlong Penalty**: Add an explicit length penalty term for the last 4k tokens (as used in DAPO) to actively discourage verbose responses
- **w/ Overlong Penalty and Robust Verifier**: Further change to a more sophisticated DeepScaleR verifier to reduce false negatives (correct solutions misclassified as incorrect)



It shows differences after 2k steps training

Outline

- Background
- JustRL Recipe
- Experiments
- **Discussion**

Discussion

➢ What this tells us

- **Not all standard tricks transfer**: The overlong penalty works in DAPO's context but hurts in ours. Techniques aren't universally beneficial; they interact with other design choices in complex ways
- **Simpler isn't always easier to improve**: We tried two seemingly reasonable modifications and both made things worse. This suggests our base recipe is achieving some balance that's easy to disrupt

Discussion

➢ What we don’t know

- We demonstrate that JustRL works well, but can't isolate why
- Is it the hyperparameters? The training dataset? The verifier design?
- Limited to two backbones in mathematical reasoning at 1.5B scale

➢ When might complexity help

- Additional techniques may be valuable under some constraints
- Establish simple yet strong baselines first, then add complexity only when you identify specific problems it solves

Conclusion

Aman @inceptmyth

really cool paper. short, sweat and answers some of the problems I had post training small LLMs with RL.

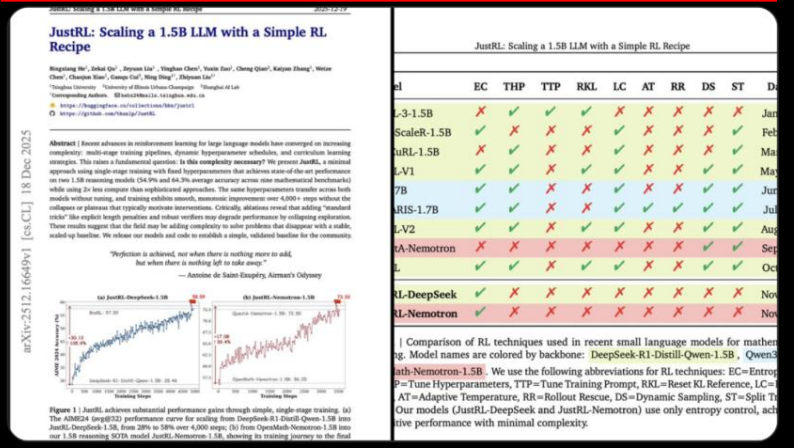
TLDR:

1) simple RL, with entropy control (asymmetric PPO clipping) works well for small LLMs. (more number of steps to train on is crucial as well)

2) no need of adaptive temperature sampling, excessive number of rollouts, length penalty rewards, Reset KL Reference, curriculum training on context length size, etc.

To my fellow researchers: if you don't have any ideas read this paper and there are lot of open problems/ experiments that the authors are urging for other folks to explore.

(P..S I love the comparison table).


JustRL: Scaling a 1.5B LLM with a Simple RL Recipe

JustRL: Scaling a 1.5B LLM with a Simple RL Recipe

	EC	THP	TTP	RKL	LC	AT	RR	DS	ST
L-3-1.5B	✗	✓	✓	✓	✗	✗	✗	✗	✗
ScaleR-1.5B	✓	✗	✗	✗	✓	✗	✗	✗	✓
uRL-1.5B	✗	✓	✗	✗	✓	✗	✗	✗	✓
L-V1	✓	✓	✗	✓	✓	✗	✗	✓	✓
7B	✓	✓	✗	✗	✓	✗	✗	✓	✓
RIS-1.7B	✓	✓	✗	✗	✓	✓	✓	✓	✓
L-V2	✓	✓	✗	✓	✓	✗	✗	✓	✓
A-Nemotron	✗	✗	✗	✗	✗	✗	✗	✓	✓
L	✓	✓	✗	✓	✓	✗	✗	✓	✓
RL-DeepSeek	✓	✗	✗	✗	✗	✗	✗	✗	✗
RL-Nemotron	✓	✗	✗	✗	✗	✗	✗	✗	✗

Comparison of RL techniques used in recent small language models for mathem
g. Model names are colored by backbone: DeepSeek-R1-Distill-Qwen-1.5B , Qwen3
ath-Nemotron-1.5B . We use the following abbreviations for RL techniques: EC=Entrop
P=Tune Hyperparameters, TTP=Tune Training Prompt, RKL=Reset KL Reference, LC=
AT=Adaptive Temperature, RR=Rollout Rescue, DS=Dynamic Sampling, ST=Split T
Our models (JustRL-DeepSeek and JustRL-Nemotron) use only entropy control, ach
tive performance with minimal complexity.

11:34 PM · Dec 25, 2025 · **17K** Views

David Wall @DavidWall9987 · Dec 23

If a 1.5B model reaches SOTA with a single RL recipe,

you have to ask whether our RL pipelines were fighting real problems — or the noise we added ourselves.

1 · 1.4K

Luco @luco_shrm · Dec 24

We seem to have passed a phase transition where less will be more. Interesting implications

105

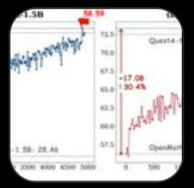
Yam Peleg @Yampeleg · Nov 14, 2025

MORE OF THIS.

TL;DR

Training small reasoning models with RL has become a race toward complexity, using multi-stage pipelines, dynamic schedules, and curriculum learning. We ask: Is this complexity necessary? We show that **JustRL**, a simple recipe with fixed hyperparameters, achieves state-of-the-art performance on two different 1.5B base models (54.5% and 64.3% across 9 math benchmarks) while using 2× less compute than sophisticated approaches. The same hyperparameters transfer across both models without tuning, and training remains stable over thousands of steps without intervention. This suggests the field may be adding complexity to solve problems that disappear with a stable, scaled-up baseline. We open-source our models and evaluation scripts as validated starting points for practitioners and researchers.

Bingxiang He @HBX_hbx · Nov 12, 2025



✨What if the simplest RL recipe is all you need?

Introducing JustRL: new SOTA among 1.5B reasoning models with 2× less compute.

...

1 · 20 · 3.2K

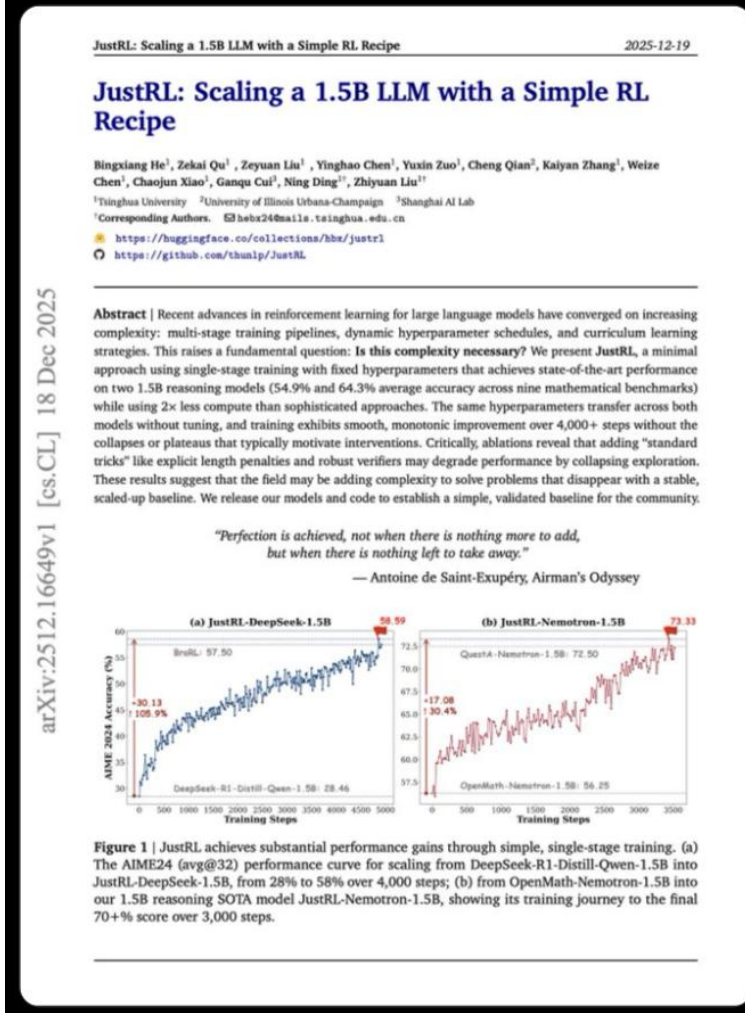
alphaXiv @askalphaxiv

Simple RL is all you need for Small LLMs

This paper shows that a single simple RL recipe can push 1.5B models to SoTA reasoning with half the compute

Suggesting whether today's complex RL pipelines are solving real problems or ones we created ourselves..

trending on alphaXiv📈


JustRL: Scaling a 1.5B LLM with a Simple RL Recipe 2025-12-19

JustRL: Scaling a 1.5B LLM with a Simple RL Recipe

Bingxiang He[1], Zekai Qu[1], Zeyuan Liu[1], Yinghao Chen[2], Yuxin Zuo[1], Cheng Qian[2], Kaiyan Zhang[1], Weize Chen[2], Chaojun Xiao[2], Ganqu Cui[3], Ning Ding[1†], Zhiyuan Liu[1†]

[1]Tsinghua University [2]University of Illinois Urbana-Champaign [3]Shanghai AI Lab

†Corresponding Authors. hebx24@mails.tsinghua.edu.cn

https://huggingface.co/collections/hbx/justrl

https://github.com/thunlp/JustRL

arXiv:2512.16649v1 [cs.CL] 18 Dec 2025

Abstract | Recent advances in reinforcement learning for large language models have converged on increasing complexity: multi-stage training pipelines, dynamic hyperparameter schedules, and curriculum learning strategies. This raises a fundamental question: **Is this complexity necessary?** We present **JustRL**, a minimal approach using single-stage training with fixed hyperparameters that achieves state-of-the-art performance on two 1.5B reasoning models (54.9% and 64.3% average accuracy across nine mathematical benchmarks) while using 2× less compute than sophisticated approaches. The same hyperparameters transfer across both models without tuning, and training exhibits smooth, monotonic improvement over 4,000+ steps without the collapses or plateaus that typically motivate interventions. Critically, ablations reveal that adding "standard tricks" like explicit length penalties and robust verifiers may degrade performance by collapsing exploration. These results suggest that the field may be adding complexity to solve problems that disappear with a stable, scaled-up baseline. We release our models and code to establish a simple, validated baseline for the community.

"Perfection is achieved, not when there is nothing more to add, but when there is nothing left to take away."

— Antoine de Saint-Exupéry, Airman's Odyssey

(a) JustRL-DeepSeek-1.5B (b) JustRL-Nemotron-1.5B

AIME 2024 Accuracy (%) Training Steps

Figure 1 | JustRL achieves substantial performance gains through simple, single-stage training. (a) The AIME24 (avg@32) performance curve for scaling from DeepSeek-R1-Distill-Qwen-1.5B into JustRL-DeepSeek-1.5B, from 28% to 58% over 4,000 steps; (b) from OpenMath-Nemotron-1.5B into our 1.5B reasoning SOTA model JustRL-Nemotron-1.5B, showing its training journey to the final 70+% score over 3,000 steps.

4:11 AM · Dec 23, 2025 · **59.3K** Views

Conclusion

If simplicity is sufficient more often than current practice assumes, that seems worth paying attention to!

🤗 Models: https://huggingface.co/collections/hbx/justrl

💻 Repo: https://github.com/thunlp/JustRL

𝕏 Thread: https://x.com/HBX_hbx/status/1988474153436090776

✨ Paper Link: https://arxiv.org/abs/2512.16649

"Non sunt multiplicanda entia sine necessitate"

——— Occam's razor

Perhaps the novelty of this work lies in the absence of novelty

THANKS

Q & A

Bingxiang He | THUNLP | Advisor: Prof. Zhiyuan Liu

Homepage: https://hbx-hbx.github.io/

2026.02.03

We are actively working on scalable RL!