# Week 9: Why RL x LLM Works

**BEH Chuen Yang**

## Abstract

This report pertains to the apparently unlikely success of Reinforcement Learning in post-training large language models (RL x LLM). We first briefly outline the differences between two common approaches to RL x LLM: Reinforcement Learning with Human Feedback (RLHF) and Reinforcement Learning with Verifiable Rewards (RLVR). Then, we propose a few hypotheses on why RL x LLM is so successful despite the multitude of challenges that should at first glance plague RL x LLM. It is hoped that this report informs readers intuitions on RL x LLM, and provides a foundation for further exploration of the topic.

## 1 Introduction

Ever since Ouyang et al. (2022) demonstrated the effectiveness of Reinforcement Learning with Human Feedback (RLHF) in post-training large language models (LLMs), there has been a surge of interest in applying RL to LLMs.

Since then, RL has culminated in many technical marvels such as the Claude 4 Family (Anthropic (2025)), Tulu 3 (Lambert et al. (2025)) and DeepSeek-R1-Zero (DeepSeek-AI et al. (2025)), with the latter even being able to omit the need for expensive Supervised Fine-Tuning (SFT) data, which is difficult to scale and requires significant human effort to collect and curate.

However, this apparent success of RL in LLM post-training is extremely surprising in and of itself. Therefore, this report aims to explore the reasons why, **despite the multitude of challenges that RL should, in theory, face when applied to LLM post-training, it nevertheless succeeds with aplomb in reality.**

## 2 Types of RL x LLM

Before we explore the mysterious success of RL x LLM, we will first explore two common approaches to RL x LLM which have been applied with mainstream success:

### 2.1 Reinforcement Learning with Human Feedback (RLHF)

As mentioned in Beh (2025), RLHF (Christiano et al. (2017)) attempts to steer an LLM's output distribution to one whose outputs are more desirable to humans. RLHF achieves this by training a reward model (RM) (a Bradley-Terry model (Bradley & Terry (1952)) parameterized by a neural network) to predict the relative desirability of any output from the LLM, then performing RL in an armed bandit fashion (Sutton & Barto (2018)) to induce the LLM to output more desirable outputs.

This approach distinguishes itself from other approaches to RL x LLM by its use of human feedback to train the RM, which technically allows the RM to be trained on a wide variety of tasks, as long as there is some way to rank the outputs of the LLM on that task.

It also distinguishes RLHF from SFT (Ouyang et al. (2022)) since the RM allows for a certain degree of generalization; that is, the LLM can be trained on a wider variety of outputs than would be possible with SFT alone.

While this is a strength of RLHF, the use of a neural network as the reward model also means that the rollout process can be quite expensive, as the RM must be queried for every output generated by the LLM. Moreover, as a mere approximation, the RM can also be inaccurate, which can lead to suboptimal policies being learned as a result of RLHF.

### 2.2 Reinforcement Learning with Verifiable Rewards (RLVR)

Again as mentioned in Beh (2025), RLVR is a more recent approach to RL x LLM pioneered by Lambert et al. (2025) and then further developed by DeepSeek-AI et al. (2025). In contrast to RLHF, RLVR attempts to steer an LLM's output distribution **not** towards human preference, but instead towards a distribution whose outputs are verifiably correct according to some task-specific criteria.

This simplicity of RLVR allows it to easily deliver results when applied to a very small subset of tasks, such as coding, mathematics, formatting, and reasoning tasks. (Lambert et al. (2025); DeepSeek-AI et al. (2025)). Indeed, this removes a lot of the usual headaches that come with having to perform RL, such as reward modelling decisions (OpenAI (2016)) and problem diversity.

However, RLVR is consequentially much more limited in scope than RLHF, and while effective, it is therefore less wieldy as a post-training technique. For example, you cannot use RLVR to make an LLM less harmful, for "harmfulness" contains many dimensions which cannot be reduced to a simple number, or a set of black-and-white criteria.

## 3 Why RL x LLM Seems Rather Difficult

It is known that solving high-dimensional problems using RL is notoriously difficult (Sutton & Barto (2018); Jones (2021)), especially when considering the set of all possible LLM outputs over today's truly monstrous context window sizes and vocabularies (Anthropic (2025); DeepSeek-AI et al. (2025); et al (2025)). Taking LLMs specifically into consideration, this is because the set of possible states and actions both grow *at least* exponentially in the context window size. [1]

This compounds to the already monumental difficulty of getting distinguishable advantages between different outputs in both RLHF (which in its base form uses some variation of the Bradley-Terry model (Bradley & Terry (1952))) as well as DeepSeek-R1-Zero's take on RLVR (DeepSeek-AI et al. (2025)), for the model could output thousands of tokens and receive but a small reward signal in the interval $[-1, 1]$.

As Salimans & Chen (2018) have empirically demonstrated, even relatively advanced RL algorithms such as Proximal Policy Optimization (PPO) (Schulman et al. (2017)) have a difficult time learning a good policy within reasonable wall-clock times when reward signals are sparse / delayed, and / or when the state and action spaces are high-dimensional.

### 3.1 Aside: KL Divergence Penalty

Moreover, RL x LLM techniques often utilise a KL divergence penalty to disincentivize the RL post-trained LLM from deviating too far from the base LLM (Ouyang et al. (2022); Lambert et al. (2025); DeepSeek-AI et al. (2025)).

While there are clear motivations for doing so (i.e. preserving the generation quality of the base LLM and preventing catastrophic forgetting), the KL divergence penalty **could be expected to** act against the RL post-training process when it comes to exploration, as it discourages the RL post-trained LLM from exploring diverse outputs that may be necessary to learn a good policy.

[1]For some intuition, consider the simplest case, where LLM generation is treated as an armed bandit problem (Sutton & Barto (2018)). With an input sequence of length $n$, and an output sequence of length $m$. If the vocabulary set is $V$, then we have $|V|^n$ possible input states, and $|V|^m$ possible output actions.

This is another factor which exacerbates the apparent difficulty of RL x LLM.

## 4 Why Does RL x LLM Work Anyway?

Yet, despite the above challenges, RL post-training of LLMs has been shown to work in practice (Ouyang et al. (2022)). Most surprisingly, RL post-training has been increasingly favored by frontier labs (Anthropic (2025); Lambert et al. (2025)), and shown to succeed under even harsher conditions, such as in DeepSeek-R1-Zero (DeepSeek-AI et al. (2025)), where RL post-training is done without any SFT data to serve as a backbone, and where the optimization algorithm used (GRPO) discards an important variance reduction technique in classical RL: advantage estimation (Schulman et al. (2017); Sutton & Barto (2018)) using a critic network.

We attempt to explain why in the following subsections.

### 4.1 Lower Reward Variance Due to Good Initialization

The best possible rationalization for the success of RL post-training is to think of pre-training as a sophisticated, high-quality initialization procedure for the RL post-training process.

Another aspect of the RL optimization process that makes RL unwieldy for tasks with sufficient complexity is its empirical sensitivity to weight initialization (Jones (2021); Andrychowicz et al. (2020)).

By pre-training on a massive corpus of text data, the base LLM is able to first figure out the structure of language, which significantly prunes the set of possible outputs to a much smaller, sensible set of token sequences. In other words, the base LLM learns a good initialization for the RL post-training process through pre-training.

Hence, the base LLM's action space in terms of the RL post-training process becomes much smaller. Not only is this more practical to explore, but it also means that the reward variance is much lower, making the problem of learning a good policy much easier than would be suggested by a simple numerical analysis of the action space size.

### 4.2 Improvements in Base Language Models

Pre-trained language models used in RL post-training have also improved significantly over the years. From the first base models like GPT-2-XL (Radford et al. (2019)) to modern base LLMs of comparable size like Qwen-2.5-1.5B (Qwen et al. (2025)), base LLMs have become much more performant across a wide variety of tasks (few-shot) (Fourrier et al. (2024)) like Instruction-Following (Zhou et al. (2023); Suzgun et al. (2022)), Language Understanding (Wang et al. (2024)), Mathematics (Hendrycks et al. (2021)), question-answering (Rein et al. (2023); Suzgun et al. (2022)), and even reasoning (Hendrycks et al. (2021); Sprague et al. (2024)). [2]

Since it is already believed that the base LLM can be thought of as a high-quality initialization for the RL post-training process, the improvements in base LLMs themselves would rather straightforwardly entail greater ease (and indeed, performance) in the RL post-training process.

### 4.3 KL Divergence Penalty Creates Stable Exploration

Considering what was mentioned in the previous section, this hypothesis may come across as a bit counterintuitive. However, Vieillard et al. (2020) find that KL regularization, when

[2]While common reasons for this phenomenon include improvements in architecture design, training data quality and quantity, and training techniques, it is worth noting that newer base LLMs may also have been trained on text chunks generated by other LLMs doing reasoning tasks. It is possible that some part of base LLMs' performance improvements come from "cheating" in this sense.

used with RL algorithms like Value Iteration (Bellman (1957)), can actually help to stabilize exploration in RL.

Where RL x LLM in the armed-bandit case is concerned, the authors notice that KL regularization helps to average (and therefore mitigate) any bias or random error in the value function updates, resulting in a more stable exploration process.

However, it is worth noting that the exact mechanisms by which this occurs are not fully understood, especially because the authors' assumptions do not hold up when neural network approximators are used. Nevertheless, it is possible that their hypothesis in the simple case still holds up in the context of Deep RL.

## References

Marcin Andrychowicz, Anton Raichuk, Piotr Stanczyk, Manu Orsini, Sertan Girgin, Raphael Marinier, Leonard Hussenot, Matthieu Geist, Olivier Pietquin, Marcin Michalski, Sylvain Gelly, and Olivier Bachem. What matters in on-policy reinforcement learning? a large-scale empirical study, 2020. URL https://arxiv.org/abs/2006.05990.

Anthropic. System card: Claude opus 4 & claude sonnet 4, 2025. URL https://www-cdn.anthropic.com/4263b940cabb546aa0e3283f35b686f4f3b2ff47.pdf.

Chuen Yang Beh. Week 7: Comparing base models and instruct-tuned models, 2025. URL ../wk7/wk.pdf.

Richard Bellman. *Dynamic Programming*. Princeton University Press, 1957. ISBN 978-0-486-42809-3. Dover paperback edition (2003).

Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952. ISSN 00063444, 14643510. URL http://www.jstor.org/stable/2334029.

Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences, 2017. URL https://arxiv.org/abs/1706.03741.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui

Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Gemini Team et al. Gemini: A family of highly capable multimodal models, 2025. URL https://arxiv.org/abs/2312.11805.

Clementine Fourrier, Nathan Habib, Alina Lozovskaya, Konrad Szafer, and Thomas Wolf. Open llm leaderboard v2, 2024. URL https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset, 2021. URL https://arxiv.org/abs/2103.03874.

Andy L. Jones. Debugging reinforcement learning. https://andyljones.com/posts/rl-debugging.html, 2021.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tulu 3: Pushing frontiers in open language model post-training, 2025. URL https://arxiv.org/abs/2411.15124.

OpenAI. Faulty reward functions, 2016. URL https://openai.com/index/faulty-reward-functions/.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners, 2019. URL https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf. OpenAI Blog.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof q&a benchmark, 2023. URL https://arxiv.org/abs/2311.12022.

Tim Salimans and Richard Chen. Learning montezuma's revenge from a single demonstration, 2018. URL https://arxiv.org/abs/1812.03381.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Zayne Sprague, Xi Ye, Kaj Bostrom, Swarat Chaudhuri, and Greg Durrett. Musr: Testing the limits of chain-of-thought with multistep soft reasoning, 2024. URL https://arxiv.org/abs/2310.16049.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018. URL http://incompleteideas.net/book/the-book-2nd.html.

Mirac Suzgun, Nathan Scales, Nathanael Schaerli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them, 2022. URL https://arxiv.org/abs/2210.09261.

Nino Vieillard, Tadashi Kozuno, Bruno Scherrer, Olivier Pietquin, Rémi Munos, and Matthieu Geist. Leveraging the average: an analysis of kl regularization in reinforcement learning, 2020. URL https://arxiv.org/abs/2003.14089.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark, 2024. URL https://arxiv.org/abs/2406.01574.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models, 2023. URL https://arxiv.org/abs/2311.07911.