Augmenting Large Reasoning Models with Contrastive Goal-Conditioned Reinforcement Learning

Devan Shah
Princeton University
ds6237@princeton.edu

Kevin Wang
Princeton University
kw6487@princeton.edu

David Yan
Princeton University
dy2617@princeton.edu

1 Abstract

As the paradigm in artificial intelligence shifts from pre-training scaling laws toward test-time training with RL, reasoning models have emerged as the next frontier. Ever since the release of DeepSeek-R1 (DeepSeek-AI et al., 2025), a growing trend involves training on complex reasoning tasks with verifiable outcomes (i.e. reward of 1 if the solution is correct and 0 otherwise, with simple format rewards). We make the observation that this outcome-oriented reward paradigm is effectively a goal-conditioned setup. Meanwhile, in the broader RL community, recent self-supervised RL algorithms have shown strong success on classical goal-conditioned settings (Eysenbach et al., 2023), where sparse reward only provides a single bit of reward feedback for each trajectory. A core question thus arises: given this quasi-goal-conditioned paradigm in NLP, can these same goal-conditioned self-supervised RL methods be used to advance LLM reasoning?

2 Introduction

Large Language Models (LLMs) trained to reason using reinforcement learning (RL) have led a paradigm shift from pre-training compute scaling to inference time compute scaling (DeepSeek-AI et al., 2025; Muennighoff et al., 2025). The *reward models* (RMs) that evaluate output quality and guide training are therefore a critical piece of modern reasoning model training.

Existing frontier reasoning models are predominantly trained with outcome-based reward models (ORMs) that consist of a numerical verifier for complex math and coding tasks (DeepSeek-AI et al., 2025). The simplicity of the RM enables training on large corpora of verifiable problems without requiring any human feedback or labeling. However, ORMs provide a sparse reward structure and only provide feedback at the end of completions. Meanwhile, process reward models (PRMs) (Lightman et al., 2023) solve this problem by giving feedback at intermediate reasoning steps, thereby providing a dense reward structure. However, PRMs either require human-annotated trajectories, which are not scalable, or language-model based annotations that are noisy.

Drawing from the success of contrastive representation learning in traditional RL tasks (Eysenbach et al., 2023), we propose a novel technique for the self-supervised learning of a contrastive reward model, allowing the benefits of dense feedback signals while only requiring problems paired with a numerical verifier for the training of both the reward model and the reasoning model.

We test our approach on two leading open-weight large language models, Qwen 2.5-7B (Yang et al., 2024) and Llama 3.1-8B (Grattafiori et al., 2024) on the game of countdown and investigate the challenges of contrastive representation learning for reward models. Code is available at github.com/dshah02/ContrastiveReasoners.

3 Related Work

Reinforcement Learning for LLMs.

To ensure language models accurately follow the preferences of users and can answer user queries beyond responding with the most likely next token, language models often undergo reinforcement learning-based post-training after self-supervised training on a language corpus to align better with human preference.

Language models can be treated as actors in a Markov Decision Process, with the prior input constituting their state and the choice of next token being the relevant actions. To incentivize behavior aligned with human desires, these language models can then receive a reward corresponding to the quality of their outputs, and using traditional methods from reinforcement learning, this reward can be used to update the weights of the language model

to ensure better alignment with human feedback.

As human-provided grading is unscalable, the most common approach for aligning language models is known as Reinforcement Learning from Human Feedback (Ouyang et al., 2022), whereby human-annotated data on the quality of text, as aligned with human preferences, is collected on a wide variety of input text, and a separate reward language model is trained to predict, for any piece of text, the expected score it would receive from a human annotator. By prompting the base language model on a variety of queries, this reward model can be used to provide feedback on the policy, and when combined with Proximal Policy Optimization (PPO) (Schulman et al., 2017), will lead to a language model that retains the language knowledge of the base pre-trained model while being more likely to earn favorable scores from human reviewers.

However, for questions with concrete verifiable answers, such as mathematics and factual problems, the reward signal does not need to be provided from a reward model, but can instead be based on whether the model correctly answered the question. Shao et al. (2024); DeepSeek-AI et al. (2025) showed that training a language model based on this verifiable reward, leveraging Group Relative Policy Optimization (GRPO) (Shao et al., 2024), can lead to models with impressive performance on complex tasks and lead to the emergence of reasoning behavior. However, note that as only a single reward is provided per model completion, the aforementioned reward techniques are extremely sparse and thus perhaps computationally inefficient.

To provide greater signal, other practitioners employ process reward models, which aim to provide a reward on each token generated rather than on the entire completion (Nath et al., 2024). Although this reward is more dense, it is also much more challenging to train, as it can be difficult to evaluate each new token without the knowledge of what future tokens may be. PRMs currently require labeled output trajectories to train a reward model on distinguishing which token-level outputs are more likely to lead to a goal with the desired label (such as "Correct"); however, training these PRMs proves challenging as human-annotated reasoning trajectories are costly to obtain and LLM-annotated data is often faulty.

Contrastive Reward Models

Hejna et al. (2024); Nath et al. (2024) propose a procedural reward model that, although requiring human-labeled trajectories, provides a different approach to training procedural reward models based on learning trajectory embeddings. Rather than predicting a reward for each subset of text, Nath et al. (2024) propose learning a feature embedding ϕ from text to $\mathbb{R}^n$ that preserves that text generated from the same trajectory are likely to map near each other. In this manner, a reward signal can be provided at the token-level by determining whether, at each step, the chosen action leads to an embedding closer to the goal embedding, with the goal embedding chosen as an average of the embeddings of successful trajectory embeddings during training.

Following (Nath et al., 2024), at a more technical level, with a dataset $\mathcal{D}$ of inputs x paired with correct trajectories y_w and incorrect trajectories y^ℓ, we will learn a reward function $r(x, y) = r'(\phi(y_T | y_{\{0,\dots,T-1\}}, x))$, consisting of $\phi : \text{text} \to \mathbb{R}^n$ and $r' : \mathbb{R}^n \to \mathbb{R}^1$. To learn the feature embedding ϕ and reward mapping r', we then aim to minimize a combined loss $\mathcal{L}^R + \lambda \mathcal{L}^C$, where

$$\mathcal{L}^R = -\frac{1}{|\mathcal{D}|}\mathbb{E}_{(x,y^w,y^\ell)\sim\mathcal{D}} \log(\sigma(r(x, y^w) - r(x, y^\ell))$$

incentivizes rewarding correct trajectories and

$$\mathcal{L}^C = \mathbb{E}_{(x,y_{0,t},y_g^+,y_g^-)} \log \left(\frac{\sigma(f(x, y_{\{0,\dots,t\}}, y_g^+)}{1 - \sigma(f(x, y_{\{0,\dots,t\}}, y_g^-)} \right)$$

incentivizes the model to correctly embed trajectories in-progress near their end-state and away from the end-state of other trajectories. The function $f(x, y_{0,\dots t}, y_g)$ is the cosine similarity between $\phi(x, y_g)$ and $\phi(x, y_{0,\dots,t})$, and $y_{0,\dots t}$ is a randomly chosen prefix of y_g^+, while y_g^- is from a different trajectory with the same prompt (Nath et al., 2024).

Countdown Task and LongProc

To measure reasoning performance on a verifiable task, we choose the countdown problem (Ye et al., 2025; Yao et al., 2023). Each countdown problem consists of a list of 4 numbers and a target number to create based on those 4 numbers and the 4 basic arithmetic operations $+, -, \times, \div$. For instance, an example might be $[3, 5, 2, 6]$ with a goal of creating 8, and a correct answer is $(6-2)\times(5-3)$. Another correct answer is $(6 \times 3) - (5 \times 2)$.

We leverage the LongProc (Ye et al., 2025) countdown benchmark in order to construct example problems and train our model. We consider the easiest set of LongProc Countdown problems, which predominantly consists of tasks requiring addition and subtraction and can be solved by procedural generation software in less than 500 tokens. Each prompt showcases an example of correctly answering a countdown problem and a description of how the model should attempt the problem.

The countdown problem has been used as a classic example of a reasoning task (Yao et al., 2023) and harder examples can be quite challenging for many people.[1]

As a reference for baseline model performance, under the LongProc Countdown prompting method, which has models attempt the problem via breadth-first search with 500 tokens, Llama 3.1-8B (Instruct) answers the easiest set of problems with 8% accuracy and Qwen 2.5-7B (Instruct) achieves 32% accuracy. For our testing, we permit the models 1024 tokens and modify the prompt, so although our results are not directly comparable, this serves as a baseline to gauge problem difficulty.

4 Approach

4.1 Contrastive Critic Design

Our primary goal is to train a critic model to understand whether an intermediate state in a reasoning trajectory is "on the right track" towards achieving the final goal. We want the critic to learn to map (intermediate state, current action) pairs and final goals into a shared embedding space. In this learned space, the embedding of a state-action pair that is part of a successful trajectory leading to a specific goal should be "close" to the embedding of that goal. Conversely, it should be "far" from embeddings of irrelevant goals or goals corresponding to different problems. To achieve this, we employ a contrastive training setup, specifically the InfoNCE (van den Oord et al., 2019) loss. This approach allows for dense feedback without requiring explicit step-by-step human labels, relying instead on the final outcome and the trajectory structure. Our design is also motivated by the work of (Nath et al., 2024), although we strive to make our method purely self-supervised.

[1]If you want to challenge yourself, try to make 20 from $[8, 3, 7, 6]$ or to find both ways to make 8 from $[3, 4, 5, 7]$.

4.2 Text-based Critics

We experimented with several different critic model architectures, as illustrated in Figure 1. Initially, we attempted to train a model that would extract embeddings from the text output of a given trajectory. In this setup, we fine-tuned a small language model, such as Rho-Math-1B (Lin et al., 2025), to use as the critic model. We randomly sampled some action $\mathcal{A}$ as a randomly chosen line within the reasoning trace, and then considered the state $\mathcal{S}$ to be the entire reasoning prefix that preceded that line. The goals $\mathcal{G}$ were randomly sampled lines that came after the action. We extracted the goal and state-action embeddings from the activations of the language model and trained it using a standard contrastive loss. The primary issue with this method was that it had an extremely poor speed/accuracy tradeoff. Even relatively small 3B models were prohibitively slow at both train and inference time, and smaller models struggled to learn any useful signal at all.

4.3 Numerical Critics

As such, we switched to a lightweight, task-specific architecture. We observed that the core task of the critic model was merely to determine whether some intermediate set of numbers $\mathcal{X}$ was "on-the-right-track" to reach some goal set of numbers $\mathcal{Y}$. Instead of recovering embeddings from text containing numerical values, we decided to directly extract numerical values to use as inputs to our critic model.

Our first revised architecture (Figure 1B) consisted of digit embeddings layers that took in the initial array of four numbers as the state $\mathcal{S}$, a randomly sampled intermediate array as the action $\mathcal{A}$, and a randomly sampled array that comes after the action as the goal $\mathcal{G}$. The state-action pair is passed through a digit embedding and concatenated. The concatenated vector is then passed through a multi-layer projection head to obtain the state-action embedding. Meanwhile, the sampled goal is passed through a separate digit embedding to obtain the goal embedding. With these two embeddings, we train the network using the same contrastive loss.

This network was significantly faster to run due to being under 1M total parameters and achieved significantly better training accuracy. However, we found that adding the critic model had relatively little impact on RL training. By probing the critic network with sample state-action goal pairs, we

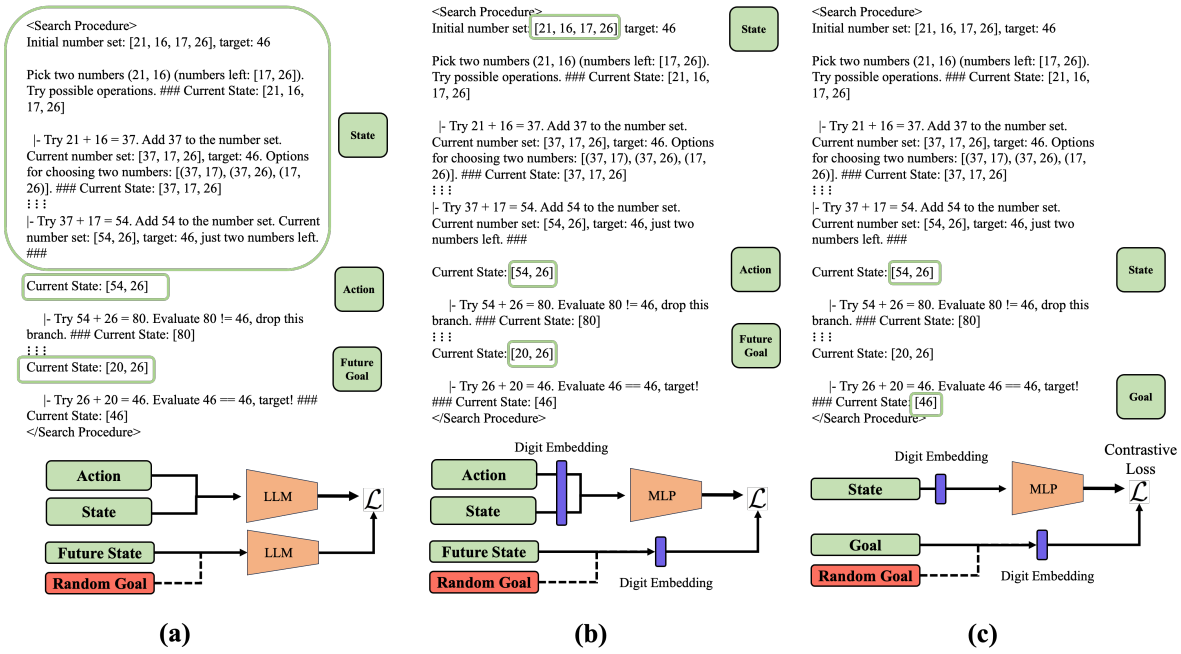

Figure 1: Overview of the three critic model architectures explored. (A) Text-based Critic: Uses a language model to extract embeddings from textual representations of state, action, and goal. (B) Numerical Critic V1: Employs digit embeddings for an initial array (state), an intermediate array (action), and a future array (goal). (C) Numerical Critic V2 (Final Architecture): Uses digit embeddings for an intermediate array (state) with the fixed final target value serving as the goal. The sampling strategy for state-action-goal triplets is shown above each respective architecture design.

found the network was not learning robust associations between the action and the goal. Instead, because the model had access to the initial set, it was circumventing the difficult task of predicting whether the intermediate action could lead to the goal and instead only learning whether the future sampled goal could be achieved with the initial four numbers. However, this representation is almost useless at inference time because the goal is *always achievable* given the initial numbers by construction. In fact, when we visualized the critic model's goal embedding space, we found that it learned a trivial even-odd partitioning of values (Figure 2).

To address this problem, we remove the initial array as input and then fix the goal $\mathcal{G}$ as the final target value, instead of randomly sampling a future state (Figure 1C). This lets the model directly learn representations from intermediate states for a single target value, which is what we desire at inference time. Intuitively, the model must learn whether target value Z can be made from some intermediate state $[A, B, C]$ (or $[A, B]$). Visualizing the goal embeddings space of this architecture reveals that the model learns a far more informative distribution of goals (Figure 3).

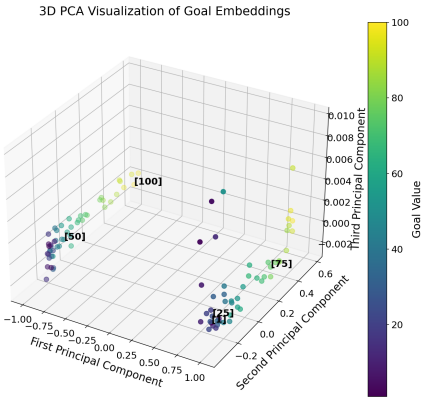

Figure 2: PCA visualization of the goal embedding space for the first numerical critic. The critic learned a trivial partitioning of values based on parity, indicating it was not capturing the desired relationships for progressing towards the target value.

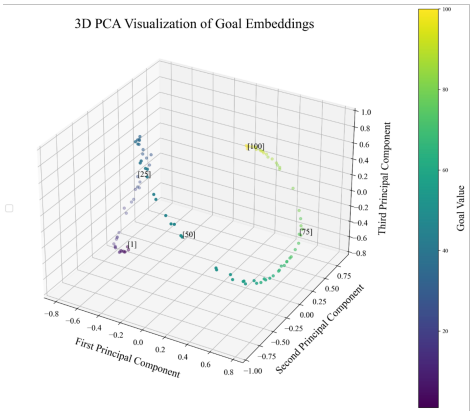

Figure 3: PCA visualization of the goal embedding space for our second numerical critic. This architecture, which uses the fixed final target value as the goal and only intermediate states as input, learns a more informative and structured distribution of goal embeddings.

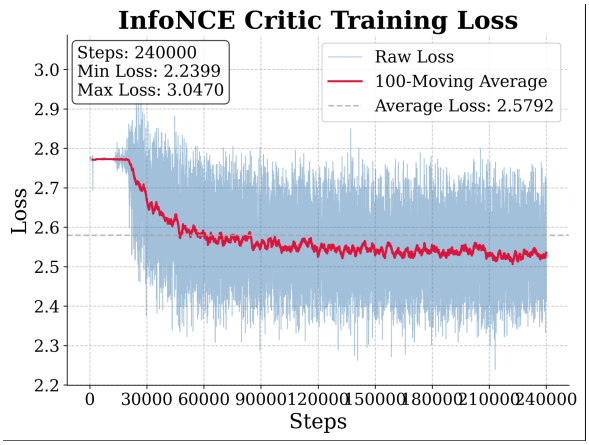

Figure 4: Training loss curve for the final critic model. The contrastive loss steadily decreases over 240,000 training steps, indicating successful convergence of the critic.

5 Experiments

5.1 Critic Model Training

We construct a large synthetic dataset of several hundred thousand unique trajectories using LongProc's procedural data generator, which creates step-by-step, natural language traces for the countdown game that correspond to a simple depth-first-search. We train our critic model on this synthetic dataset for 240000 steps with a batch size of 16 state action pairs from different trajectories. Our critic model successfully converges as seen in Figure 4. We visualize the effectiveness of the contrastive training by examining the cosine similarity matrix of state-action versus goal embeddings for a batch, as shown in Figure 5.

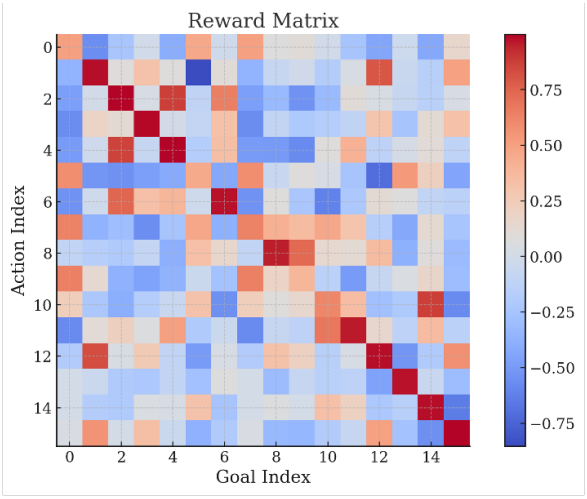

Figure 5: Heatmap of the similarity matrix between state-action embeddings and goal embeddings for a batch of 16 samples during critic model training. The strong diagonal indicates that positive pairs (state-action and goal embeddings from the same trajectory fragment) have higher similarity than negative pairs, demonstrating the effectiveness of the contrastive learning objective.

5.2 LLM Training

We run Group Relative Policy Optimization (GRPO) for the countdown task on Qwen2.5-7B and Llama-3.1-8B-Instruct to evaluate our reward model. We used the LongProc countdown dataset for training data and the LongProc countdown verifier for verification rewards. All models are trained for 1500 steps. We first trained the baseline models using only the verification outcome reward model. The Qwen model achieves approximately an 80% sucess rate. Meanwhile the Llama model struggles, rising to around 75% success rate through a less stable trajectory. This behavior is expected, because Ye et al. (2025) observed that the base Qwen model is much stronger at countdown than Llama. As such, Qwen easily achieves a extremely high success rate, while Llama suffers from higher variability when learning due to its lower initial capabilities and sparse reward feedback.

When training with our critic model feedback in addition to the standard verification reward, we find that Qwen's performance slightly to around 70% success rate while Llama decreases to about a 63% success rate. However, we note that our critic did slightly improve LLama's training in the first half of training. We hypothesize that the performance drop is because the critic is somewhat noisy, which might be a detriment once the model is at a suffi-

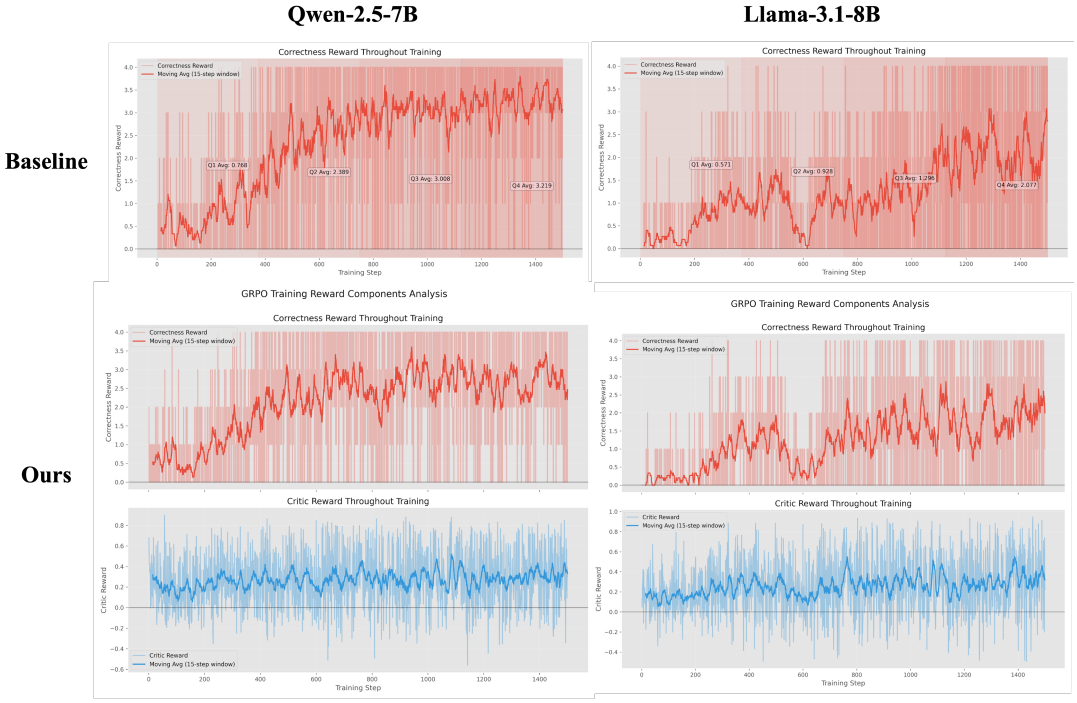

Figure 6: Group Relative Policy Optimization (GRPO) results on the countdown task for Qwen 2.5-7B (left) and Llama 3.1-8B (right). Performance is measured by success rate over 1500 training steps. The graphs compare the baseline model trained with only outcome-based rewards (GRPO Baseline) against the model augmented with our contrastive critic model (GRPO + Contrastive Critic).

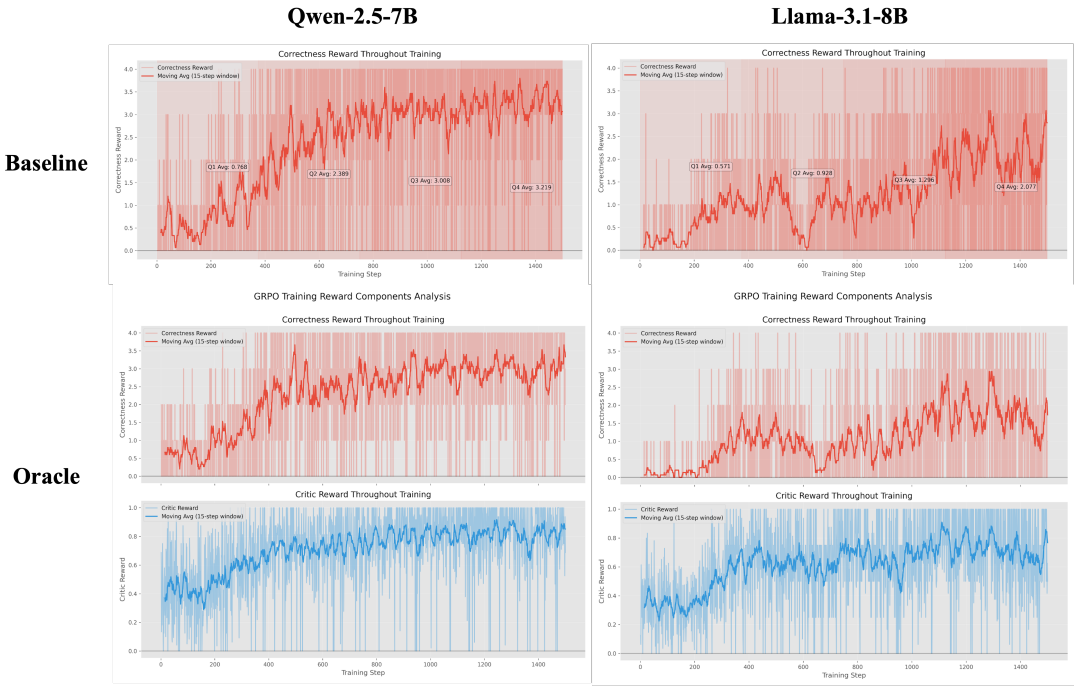

Figure 7: GRPO results on the countdown task for Qwen 2.5-7B (left) and Llama 3.1-8B (right). Performance is measured by success rate over 1500 training steps. The graphs compare the baseline model trained with only outcome-based rewards (GRPO Baseline) against the model augmented with Oracle critic feedback, where the Oracle critic uses brute-force searching to reward the model only if it makes the perfect move. We note that over the course of training, Oracle critic rewards increase, corresponding to the model's improveds searching performance.

ciently high level of performance. This would explain why the Llama model trained with our critic has slightly better performance early on, when the noisy critic model is helpful, but ultimately achieves a lower performance. These results, illustrated in Figure 6, suggest that our method might be more promising and show better results on even smaller base models that would otherwise be too weak to have convergent/stable RL training.

We additionally experiment with an oracle critic model that algorithmically computes whether the goal state is achievable, instead of learning it contrastively. We observe that the Qwen model trained with an additional oracle critic achieves better performance than the baseline (90% vs 80%), which further suggests that the noisy results of our trained critic model contributes to the decline in RL performance. The results of the experiments with the oracle model are illustrated in Figure 7.

5.3 Tools

All experimentation was done in PyTorch (Paszke et al., 2019) and our code-base is dervied from the Unsloth Llama 3.1 GRPO notebook (Unsloth.ai, 2025) and LongProc's countdown task code (Ye et al., 2025).

6 Conclusion

We have demonstrated a proof-of-concept contrastive reward model that can aid RL training of large language models on reasoning tasks. However, our method still faces several limitations. First, our critic model architecture is task-specific; ideally, the initial language model architecture should be used for the most task flexibility given sufficient compute. Second, we evaluate on a toy reasoning task for which we can generate large amounts of synthetic data for offline. A significant challenge remains developing a technique to train the critic model in conjunction with the base model on online trajectories, which will allow this method to be applied to a broader range of tasks.

References

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. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning.

Benjamin Eysenbach, Tianjun Zhang, Ruslan Salakhutdinov, and Sergey Levine. 2023. Contrastive learning as goal-conditioned reinforcement learning.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits,

Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr

Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The llama 3 herd of models.

Joey Hejna, Rafael Rafailov, Harshit Sikchi, Chelsea Finn, Scott Niekum, W. Bradley Knox, and Dorsa Sadigh. 2024. Contrastive preference learning: Learning from human feedback without rl.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Yelong Shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, and Weizhu Chen. 2025. Rho-1: Not all tokens are what you need.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling.

Vaskar Nath, Dylan Slack, Jeff Da, Yuntao Ma, Hugh Zhang, Spencer Whitehead, and Sean Hendryx. 2024. Learning goal-conditioned representations for language reward models.

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. 2022. Training language models to follow instructions with human feedback.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models.

Unsloth.ai. 2025. Llama3.1 (8b)-grpo notebook. `https://colab.research.google.com/github/unslothai/notebooks/blob/main/nb/Llama3.1`.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2019. Representation learning with contrastive predictive coding.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. 2024. Qwen2 technical report.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models.

Xi Ye, Fangcong Yin, Yinghui He, Joie Zhang, Howard Yen, Tianyu Gao, Greg Durrett, and Danqi Chen. 2025. Longproc: Benchmarking long-context language models on long procedural generation.