STABILIZING TARGETED ON-POLICY GUIDANCE IN AGENTIC REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Naive on-policy RL is often bottlenecked by the capabilities of the base model, as insufficient coverage can result in an inability to learn from certain rewards. At the same time, rewards in modern post-training systems often come from highly semantic sources, e.g. judge models with a rubric of criteria. Can we utilize such structures to provide a rich directional signal that enables improving on otherwise difficult to optimize rewards? In this work, we develop a framework we refer to as targeted guidance (TGRL) that learns from per-turn textual feedback via an on-policy self-distillation objective. Our focus is not only on directly improving hard-to-optimize behaviors, but on how to do so online without degrading baseline RL performance. Over a range of settings across an agentic multi-turn software engineering environment, we find success by combining targeted self-distillation with a per-criteria RL reward, and by only constructing feedback on turns where the policy specifically fails. We further examine the design choices that lead to stable entropy and reduce oscillatory behavior throughout training.

1 INTRODUCTION

The strength of large-scale reinforcement learning (RL) has been well established in recent years. However, there is often a limit to how much a policy can improve from naive on-policy RL (Yue et al., 2025). The primary issue is that of *insufficient coverage* – even if a reward function is carefully selected to encourage desired behavior (e.g. use a new tool with the right format), the policy will nevertheless fail to improve if it does not have a chance of performing this behavior. These problems are especially prominent in agentic settings, with multi-turn tool-calling, where a policy may be required to chain together *many* complex actions in order to receive a reward signal. Fundamentally, while RL incentivizes existing good behaviors, such methods do not communicate *how to perform new behaviors*.

At the same time, rewards in modern post-training systems often come from a range of highly semantic sources, e.g. judge models provided with a rubric of stylistic criteria. Naive RL methods condense the set of such signals into a single scalar reward, incurring a significant loss of information. Rather, can we utilize this existing machinery to provide a rich source of directional signal that allows us to learn from otherwise hard-to-optimize rewards?

In this work, we describe a framework we refer to as targeted guidance (TGRL) by utilizing per-turn textual feedback via an on-policy self-distillation objective. The main idea is that each binary reward component can additionally define a textual feedback string. If a given rollout or agent turn fails to satisfy any reward criteria, then alongside the typical policy gradient loss, we minimize KL towards an expert defined as a policy with the same underlying weights, but with the given feedback placed in context. For multi-turn settings, feedback is placed directly prior to the offending agent turn.

Importantly, our aim is that such signal can directly improve targeted policy behaviors without degrading baseline RL performance. Naive on-policy distillation methods have previously been shown to cause phenomena such as reasoning shortening (Kim et al., 2026; Kaur et al., 2026; Peng et al., 2026) or diversity collapse (Jin et al., 2026; Agarwal et al., 2024). We find that by combining the standard self-distillation objective with a per-criteria standard RL reward, and by only constructing a feedback string on the turns where the policy specifically fails, we are able to stabilize the learning process and broaden the policy capabilities *while maintaining performance on primary RL tasks*.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

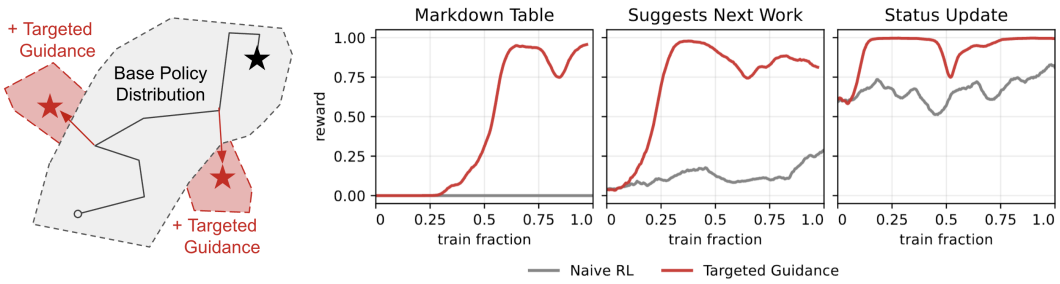

Figure 1: **Naive RL fails to optimize criterion that fall outside the support of base policies.** Exploration issues in RL can manifest as slow learning (middle) or in the worst case a complete failure to improve (left). On the other hand, targeted guidance enables policies to directionally improve specific aspects of their behavior and thus continue to learn.

We empirically examine our methodology throughout an agentic setting in which a Qwen3-Coder-30B model is trained to solve software engineering tasks through multi-turn harness involving tool calling and code editing. Notably our focus is not on improving coding pass-rate (which a well-tuned RL procedure can already do well), but rather on improving specific criterion in which naive RL procedures *fail* to improve – e.g. calling under-trained tool formats or writing with a specific style. We show that targeted guidance results in stable exploratory improvement even when multiple conflicting criterion are present, or a certain criterion must be satisfied across multiple turns. We find that by appropriately selecting the distillation weight and applying distillation to internal thinking traces, we are able to stabilize the entropy and reduce oscillations in the policy training trajectory.

2 Related Work

Exploration in deep reinforcement learning. Exploration has been a recurring theme of deep reinforcement learning, with a large body of work considering exploration from the perspectives of state-coverage (Bellemare et al., 2016; Hazan et al., 2019; Lee et al., 2019; Ecoffet et al., 2021; Liu & Abbeel, 2021), curiosity (Pathak et al., 2017; Burda et al., 2019), or skill discovery (Gregor et al., 2016; Eysenbach et al., 2019; Park et al., 2024; Ajay et al., 2021; Pertsch et al., 2020; Frans et al., 2024). A number of works have accordingly applied such methodologies to the language model RL setting (Zhang et al., 2026; Song et al., 2025; Sun et al., 2025; Dai et al., 2025; Gao et al., 2025; Li et al., 2025; Qu et al., 2026b). More uniquely to language models, the idea of exploration via strong pretrained priors has been explored in forms such as self-reflection (Shinn et al., 2023; Madaan et al., 2023; Qu et al., 2024; Kumar et al., 2025) or process rewards (Uesato et al., 2022; Lightman et al., 2024; Wang et al., 2024; Setlur et al., 2025). A solid body of work has explored how privileged information (e.g. human reference solutions) can be used as exploratory signal (Yan et al., 2025; Nath et al., 2025; Qu et al., 2026a; Fu et al., 2026; Chen et al., 2024; Zhou et al., 2026). Our work most similarly relates to the latter class of methods, in that we aim to utilize signal from textual feedback to broaden the coverage of policies. Unlike most prior settings, we do so in a manner that natively uses per-turn feedback, and generates per-token guidance.

Distillation signal from expert policies. It is well-known that strong experts can provide rich signal to improve downstream models (Hinton et al., 2015; Kim & Rush, 2016). From an RL perspective, a body of works claim rich generalization can arise when distillation signal is be generated in an *on-policy* manner (Ross et al., 2011; Agarwal et al., 2024; Gu et al., 2024; Lu & Thinking Machines Lab, 2025). In cases where a strong expert model is not available, prior works have examined the question of whether a strong policy constructed by re-using the same weights as the student, but with privileged information placed in the context (Askell et al., 2021; Snell et al., 2022; Ye et al., 2026; Lopez-Paz et al., 2016; Zhao et al., 2026; Shenfeld et al., 2026; Penaloza et al., 2026), often referred to as on-policy self-distillation (OPSD) methods. Our work builds off prior OPSD methodology and most closely relates to methods that encourage stability by gating distillation signal to failing rollouts (Li et al., 2026; Ding, 2026; Lu et al., 2026). However, while these works consider using distillation to improve a single monolithic reward, our work instead aims to improve multiple auxiliary criterion (e.g. style, tool formats, etc) while preventing degradation in baseline RL performance.

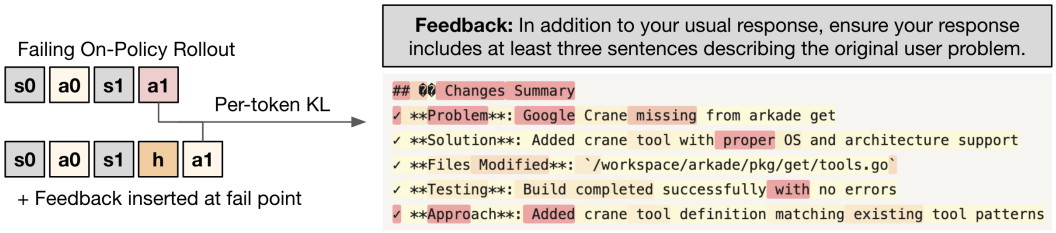

Figure 2: **Targeted guidance derives per-token signal from per-turn hints.** In cases where rollouts fail to satisfy certain criteria, e.g. to utilize a certain writing style, textual feedback is employed to guide the policy towards the solution. We instantiate this via a per-token KL loss against a feedback-conditioned expert policy.

3 Background and Preliminaries

Markov decision process. We define a Markov decision process (MDP) by a state space S, an action space A, an initial state distribution $p(s_0)$, a transition function $p(s'|s, a)$, and a reward $r(s)$. We define a parameterized policy $\pi_\theta(a|s)$ as a probability distribution of actions, which combined with the MDP creates a distribution of trajectories τ. We write the return on trajectories as $R(\tau)$.

Language models as policies. Throughout this work, we focus specifically on a policy class defined by a neural network language model. We assume access to a base model which defines a starting policy π_0. For the agentic use case, the base model is generally a pretrained language model that has gone through a supervised finetuning (SFT) phase to adapt it to the tool call setting. As this work explicitly considers multi-turn environments with agents, we abstract away from individual tokens and refer to actions as an entire set of autoregressively generated tokens in sequence, e.g. a specific tool call. We assume that the Markovian state is fully defined by the sequence of previous actions and their underlying tokens.

Rewards and judges. The reward function for this system $R(\tau)$ is nonzero only at the end of the rollout. We assume that the reward is a weighted additive function $R(\tau) = \sum_i \lambda_i R_i(\tau)$ of a prespecified set of criteria. The primary component of the reward is verification of task completion, but there are a collection of other properties such adherence to stylistic guides or use of specific tools. We will assume these rewards properties are mostly independent, e.g. it is possible to modify style without impacting task completion. While not central to the method, we primarily focus on the case of judge-based rewards, where each criterion R_i is implemented as a call to judge LLM. This judge is equipped with additional privileged information on the correct way to solve the problem, and in addition to producing reward scores, can provide textual feedback justifying its result.

4 Textual Feedback as a Targeted Directional Signal

As illustrated in Figure 1, it is often the case that naive policy gradients will fail to optimize even simple criterion, if the actions that satisfy such criterion are not within the support of the base policy. In the following section, we describe a strategy to construct a source of *directional* policy improvement signal, which communicates not which rollouts were successful, but rather *how to modify* specific aspects of a policy's behavior.

Our main approach is to utilize a per-token distillation target to provide this directional signal (Figure 2). Rather than returning a single scalar, each reward component can return *textual feedback* in the form of a feedback string f associated with a specific turn of the agent rollout. Inserting f before that turn defines an expert that shares the current policy's weights, $\mu(\cdot \mid s) = \pi_\theta(\cdot \mid s, f)$ at targeted turns and $\mu = \pi_\theta$ elsewhere. Following what is often referred to as on-policy self distillation (Zhao et al., 2026), we minimize KL between the live policy and μ over the on-policy state distribution d^π, treating μ as fixed within an update.

Expert distillation is a policy improvement operator. Following arguments from (Kakade & Langford, 2002; Schulman et al., 2015; Achiam et al., 2017), we claim that as long as the constructed

experts have a higher expected return versus the current policy:

$$\mathbb{E}_{s\sim d^{\pi}}\left[A^{\pi}(s,\mu)\right] > \mathbb{E}_{s\sim d^{\pi}}\left[A^{\pi}(s,\pi)\right], \tag{1}$$

and the KL divergence between the two remains small:

$$\mathbb{E}_{s\sim d^{\pi}}\left[\mathrm{KL}\big(\pi(\cdot \mid s)\,\|\,\mu(\cdot \mid s)\big)\right] < \epsilon, \tag{2}$$

then for sufficiently small step size, the self-distillation objective will result in an improvement over the previous policy iteration. While in practice we will not have a guarantee that both criterion above are true, we can use them as a reasonable heuristic for how to design our methodology.

Stabilizing the targeted guidance update. The aim of our proposed framework is to integrate targeted distillation-based guidance into multi-turn RL in a way that balances exploratory pressure across targeted facets while ensuring that baseline RL performance is not affected. Combining the approaches naively leads to some risk of policy collapse or stagnation. We therefore introduce two simple ideas that stabilize the training procedure, as described below.

The first main idea is that we would like to minimize the amount of unnecessary pressure from the distillation objective, which can be heuristically understood as tightening the divergence between the expert and live policy (Equation (2)). Thus, in contrast to prior work which places privileged information into a global context to construct the self-distillation target, we place the feedback string right before the affected agent turn, and mask out the KL loss on all other turns. Additionally, the distillation objective is *only evaluated on agent turns that contain a failed criterion.*

The second core idea is that while self-distillation can provide a strong directional signal of improvement, it is less clear that such objectives result in stable convergence. For this reason, our proposed methodology is to combine the distillation signal with a standard RL reward on each desired criteria. The intuition is that distillation serves as a targeted way to ensure the policy learns the relevant behavior, while it is the unbiased RL objective that enables the policy to retain and refine such behaviors.

The distillation loss in our setting is calculated via reverse KL distance between the student and the teacher. In our experiments, we found that forward-KL and Jensen-Shannon distance performed similarly. To reduce undesirable effects from inference-trainer mismatch, we utilize a full-vocabulary KL loss computed entirely on the trainer side. The combined objective is therefore:

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{GRPO}}(\theta) + \lambda \sum_{t\in\mathcal{F}} \mathrm{KL}\big(\pi_\theta(\cdot \mid s_t)\,\|\,\pi_\theta(\cdot \mid s_t, f_t)\big), \tag{3}$$

where $\mathcal{F}$ is the set of agent turns that fail a criterion, f_t is the feedback for turn t, λ is the distillation weight.

How should one design the textual feedback format? Recall that an important criterion for expert guidance to be an effective improvement operator is that the expert itself must have higher expected reward (Equation (1)). As experts in our case are defined as the same model but with certain information in the context, we can iterate on this context via a simple procedure – query the base model with a considered prompt, measure average performance against the contextless model, and only allow prompts which show an average improvement. Note that although we evaluate the feedback via autoregressive generation in this procedure, the actual application of this feedback is for per-token on-policy distillation.

Targeted guidance is asymptotically unbiased. While auxiliary signals can be necessary to solve hard exploration problems, they can at times prevent policies from converging to the optimal policy. As an example, it is possible that $\mathrm{KL}\big(\pi_\theta(\cdot \mid s_t)\,\|\,\pi_\theta(\cdot \mid s_t, f_t)\big)$ is nonzero even for an optimal policy due to spurious affects of conditioning on feedback. In the case of our proposed framework, we argue that by gating the distillation loss to only affect failing turns, such issues can be bypassed, assuming that an optimal policy exists which consistently satisfies all reward criteria.

5 Empirical Experiments with Targeted Guidance

5.1 Didactic Example: Tool Format Correction

As a motivating example, we examine an auxiliary criterion that agent policies struggle with: learning to use new tools. Oftentimes, one may want to introduce new tools during the RL phase, even

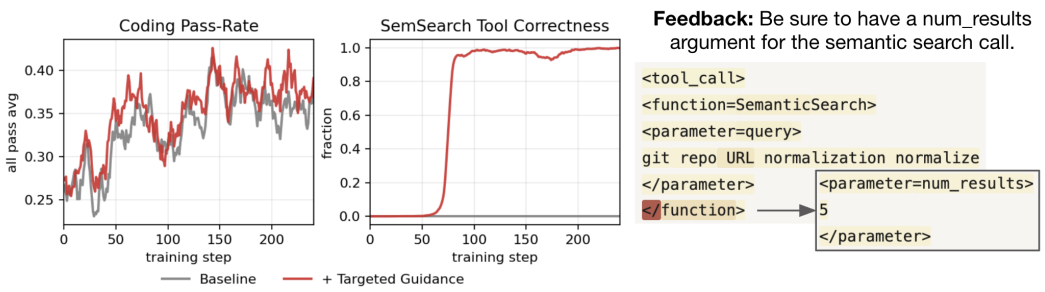

Figure 3: **Didactic example: Targeted guidance teaches new tool call formats.** In cases where the base policy fails to fulfill certain criterion, e.g. to utilize a certain tool call format, textual feedback is utilized to guide the policy towards the solution. We instantiate this via a per-token KL loss against a feedback-conditioned expert policy. As shown on the right in red, only the token in which the student and expert diverge (in this case, the tool format delimiter) receives a strong signal.

when the base model has already been through a supervised fine-tuning phase that did not include this tool. We show now that targeted guidance provides a natural way to address this problem without performance regression.

Our specific setup is the optimization of a multi-turn coding agent policy, in which an agent utilizes a series of tool calls to solve a test case and receive a primary reward. We train such a model with standard policy gradient methods, assigning each rollout a weighted sum of pass-rate, and length-penalty rewards. We consider the setting in which we are introducing a new tool-calling format to the "semantic search" tool. This format involves specifying an additional parameter (the number of lines that the search should return). Naively, the base model fails to utilize this new format at all, even when example usage is placed in the prompt.

To fix this behavior, we can utilize targeted guidance. During training, whenever the agent uses the semantic search tool without the correct format, we inject a feedback string right before the turn with the error. The policy conditioned on the feedback string then becomes an expert to serve as a source of guidance signal, which we combine with the standard policy gradient loss during RL training.

As shown in Figure 3, a policy trained with targeted guidance learns the new tool calling format, without suffering a regression on the primary coding performance. We attribute this to the guidance signal being targeted and minimal – as shown on the right, only tokens which show strong disagreement to the expert receive a large signal. Importantly, guidance is always done on-policy exactly where the error occurs, to reduce any spurious distillation effects. As the model fixes the undesired behavior, the guidance signal accordingly drops to near zero.

5.2 Can Targeted Guidance work when considering a suite of criteria?

We now examine applying targeted guidance in a more complicated setting, in which we desire to optimize a suite of criterion, some of which may conflict with each other. Specifically, we focus on the problem of improving the communication style of coding agents. We create a set of binary rubric items to be verified by an external judge, such as "include a markdown table" or "describe the user problem". For each criterion, we define a binary reward function along with an associated static feedback string.

The aim is to enable policies to reliably satisfy the stylistic criterion, while also maintaining performance levels on the primary coding tasks. In particular, our goal is that policies trained via targeted guidance should explore, while also not showing any degradation on coding pass-rate as compared to a baseline policy trained without any style criterion at all.

Our base policy is a Qwen3-Coder-30B model further SFT'd for agentic coding in our harness. In all methods below, we combine various style-optimization methodologies with a baseline GRPO algorithm that optimizes for coding pass-rate as well as constraints total output length. We train for 200 steps with a batch size of 32 groups with 32 sequences each, using Adam and a learning rate of

Table 1: A comparison of methods that aim to learn a suite of stylistic criteria (main columns), while maintaining primary coding pass-rate (right column). Errors represent standard deviation over the last 32 steps of training.

Method	Markdown	Sol. Desc	Req. Order	Approval	Bolded	Cmd Ran	Suggest Work	Pass Rate
Baseline	$0.00_{\pm.00}$	$0.70_{\pm.06}$	$0.04_{\pm.03}$	$0.00_{\pm.00}$	$0.43_{\pm.07}$	$0.00_{\pm.00}$	$0.03_{\pm.03}$	$\mathbf{0.33}_{\pm.05}$
GRPO	$0.00_{\pm.00}$	$\mathbf{0.96}_{\pm.02}$	$0.48_{\pm.08}$	$0.00_{\pm.00}$	$\mathbf{0.95}_{\pm.01}$	$0.00_{\pm.00}$	$0.20_{\pm.07}$	$\mathbf{0.33}_{\pm.07}$
OPSD	$0.77_{\pm.04}$	$0.88_{\pm.03}$	$0.29_{\pm.03}$	$0.25_{\pm.09}$	$0.90_{\pm.02}$	$0.01_{\pm.00}$	$0.44_{\pm.04}$	$0.30_{\pm.05}$
Online SFT	$0.01_{\pm.01}$	$0.78_{\pm.03}$	$0.29_{\pm.03}$	$0.02_{\pm.01}$	$0.69_{\pm.03}$	$0.00_{\pm.00}$	$0.28_{\pm.07}$	$0.26_{\pm.05}$
LaTRO	$0.00_{\pm.00}$	$0.57_{\pm.19}$	$0.03_{\pm.02}$	$0.00_{\pm.00}$	$0.21_{\pm.11}$	$0.00_{\pm.00}$	$0.07_{\pm.04}$	$0.14_{\pm.09}$
Verifree	$0.01_{\pm.02}$	$0.83_{\pm.04}$	$0.26_{\pm.14}$	$0.01_{\pm.01}$	$0.65_{\pm.19}$	$0.00_{\pm.00}$	$0.15_{\pm.11}$	$0.20_{\pm.06}$
TGRL (Ours)	$\mathbf{0.85}_{\pm.09}$	$\mathbf{0.95}_{\pm.01}$	$\mathbf{0.77}_{\pm.05}$	$\mathbf{0.55}_{\pm.06}$	$\mathbf{0.99}_{\pm.00}$	$\mathbf{0.64}_{\pm.13}$	$\mathbf{0.85}_{\pm.03}$	$\mathbf{0.33}_{\pm.05}$

3e-6. We utilize an asynchronous RL pipeline (Piché et al., 2025) where the inference policy may be older than the currently trained policy.

We compare against a variety of representative methods:

- **Baseline** represents a policy that only optimizes the standard coding reward. This serves as a baseline for the expected coding pass-rate when we do not take style criteria into account.
- **GRPO** (Shao et al., 2024) shows the performance of a standard RL policy where each stylistic criteria is assigned a binary reward, whose sum is optimized as part of the standard GRPO objective.
- **OPSD** (Zhao et al., 2026), a representative on-policy distillation method, which makes use of the per-criterion feedback to provide distillation signal. Unlike TGRL, no additional RL reward is used beyond the baseline.
- **Online SFT**, a representative iterated SFT method, in which the feedback is used to construct an explicit teacher policy which autoregressively generates alternative teacher turns to replace turns that fail a criterion. These alternative turns are then used as SFT targets for the live policy during the RL process.
- **LaTRO** (Chen et al., 2024), a representative likelihood-as-reward method. Similarly to online SFT, we construct alternative turns from a teacher. The likelihood of the alternative turns are then assigned as a reward for all previous turns in the rollout.
- **Verifree** (Zhou et al., 2026), which we instantiate as using the same framework as LaTRO except using the log-likelihood as the reward.
- **Targeted Guidance (TGRL, ours)**, in which both GRPO and distillation are combined. A distillation loss is only applied to turns in which a nonzero set of criterion are unfulfilled.

As shown in Table 1, policies trained with targeted guidance are able to improve on a wide range of evaluation criteria, while maintaining a coding pass-rate similar to that of standard GRPO. Naive OPSD maintains a relatively healthy coding pass-rate, however, the improvements on style criteria are less consistent, resulting in a worse final performance.

On the other hand, the online SFT, Verifree, and LaTRO policies result in a non-trivial degradation in coding pass-rate. In the case of online SFT, this may be because the continual SFT loss during RL limits how much the policy can change at each step, especially in an asynchronous setting. Verifree and LaTRO result in a further degradation in coding pass-rate. We hypothesize that the instability of these methods comes from the fact that unlike the on-policy methods, these three rely on *sampling entirely new actions from the feedback-conditioned expert*. While increasing the likelihood of such actions may help in terms of style, there is no guarantee that those actions are strong in terms of primary coding pass-rate. Thus, it is likely that instability arises from increasing likelihood of potentially suboptimal actions, especially in an asynchronous regime where such actions may become stale.

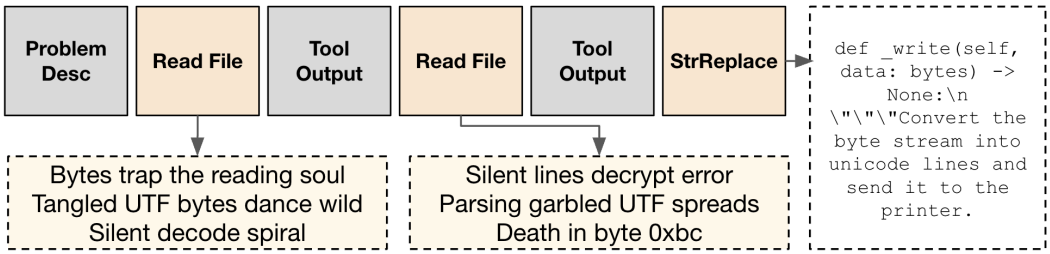

Figure 4: A haiku-style poem produced by a policy trained via targeted guidance. Feedback is inserted at every turn in which the desired criterion is not achieved.

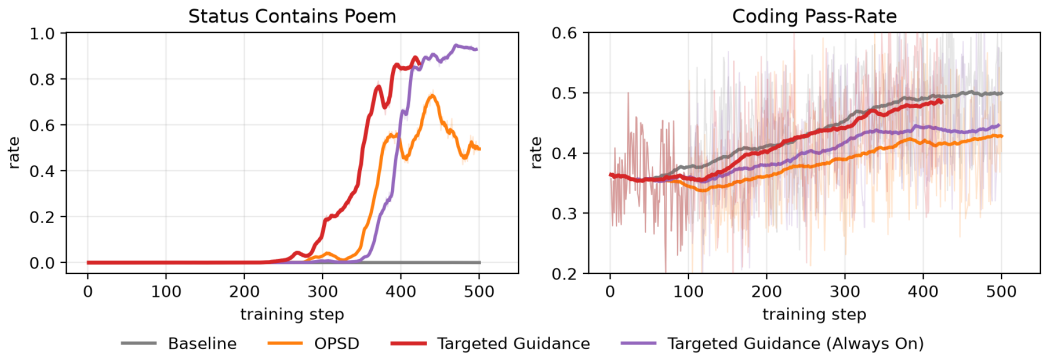

Figure 5: **Targeted guidance and alternatives as used to train models that report status updates as a poem.** The baseline GRPO run fails to optimize this out-of-support objective entirely. Naive OPSD learns, but is unstable. A variant of targeted feedback where feedback is *always* provided results in a regression in coding pass-rate.

5.3 Does Targeted Guidance remain stable even when optimizing multi-turn interleaved feedback?

The above experimental setting considered stylistic criteria on the final message style, which only occurs at the end of an agentic rollout. We are also interested in improving the behavior of non-final actions that occur throughout an agent's rollout.

To examine this, we construct a setting where *each* agent turn is graded on verifiable auxiliary criteria. To test the capability of the method, we adopt a criterion that requires significant deviation from the base model behavior: we desire the model to at each turn give a status report in the form of a poem. This is an out-of-distribution behavior that lies outside the support of the base model.

Figure 5 showcases the learning trajectories of targeted guidance and baseline methods under this setting. Note that the GRPO baseline here results in a consistent coding pass-rate improvement, as expected, but completely fails to optimize the poem reward criterion. In comparison, the policy trained with targeted guidance is able to adopt the poem output style while minimizing regression on coding performance.

Ablations on two aspects of the framework reveal their necessity. In the OPSD run, where expert distillation guidance is applied but not combined with a corresponding reward for poem-correction, the model does learn to output poems at a reasonable rate, but the policy itself is unstable and fails to consistently hold this behavior. In addition, there is a significant regression to coding pass-rate. This regression also occurs when utilizing a variant in which the poem feedback is *always* applied, regardless of whether that particular rollout was successful or not.

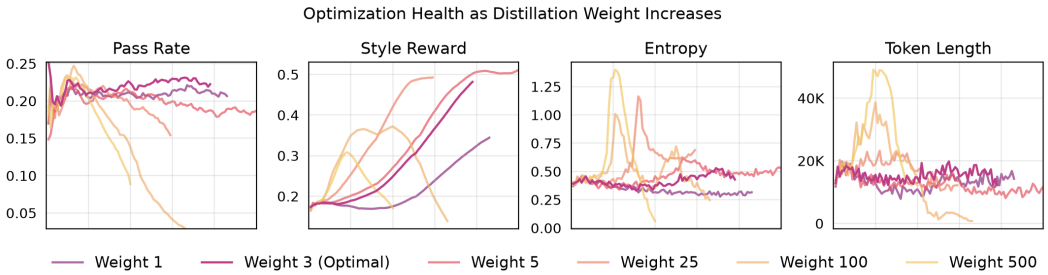

Figure 6: **Entropy spikes can occur if distillation weight is too high.** In our setting, this results in a spike in average token length, followed shortly by a collapse in pass-rate. With low enough distillation weight, the policy is stable throughout RL and style rewards can continue to improve without coding pass-rate degradation.

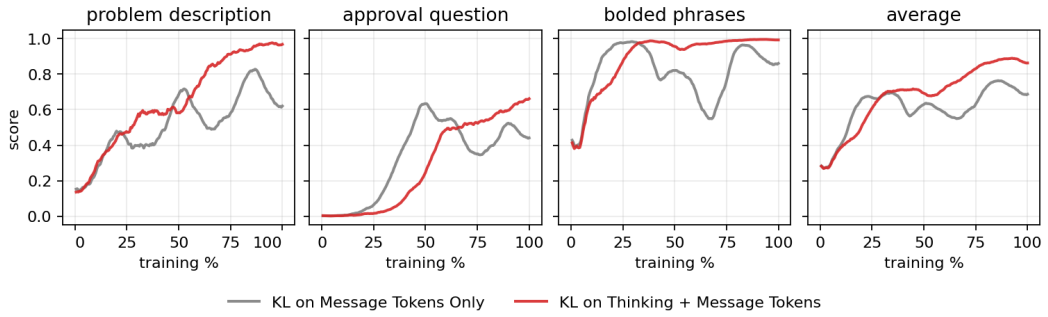

Figure 7: **Including thinking tokens in the distillation objective is crucial for stability.** In cases where the thinking is excluded from distillation signal, style criterion tend to display oscillatory behavior where certain criterion may improve, but later fall back down.

5.4 What are the failure cases of the method, and how can they be alleviated?

Stabilizing entropy. We find that an early predictor of collapse when utilizing distillation signal is a sharp increase in policy entropy. In our setting, this corresponds to an increase in the average output length, and is quickly followed by a decrease in the coding pass-rate.

An increase in entropy from exploratory signal is expected, to some degree. As by nature the behaviors that require exploratory signal are low likelihood under the live policy, increasing the likelihood of such behaviors will cause an increase in entropy. The issue is not entropy increase at all, but rather a spike in entropy increase that implies an over-correction or collapse. One possible explanation of the severity of such spikes is due to the recursive nature of self-distillation – not only does the student increase in entropy, but the expert distribution may also increase over time as both policies share the same underlying weights.

As shown in Figure 6, we can replicate high entropy collapse settings by steadily increasing the weight of the distillation signal. While such runs display a higher style reward at the start, they quickly collapse and end up failing in both pass rate and in style criterion. On the other hand, lower distillation weights do not show this type of instability at all, but in turn improve on style at a slower rate.

Stabilizing intra-reward oscillations. A phenomenon that showed up when examining the set of stylistic rewards is that the policy would oscillate between satisfying a subset of the criterion, then improve on others while degrading on others. This behavior may arise from the competing pressure for all stylistic rewards – when a distillation signal is present to push the model towards a certain style, the model may in turn be disincentivized to use a different style.

We find that such oscillatory behavior becomes much less noticeable when the distillation signal is applied also to the thinking tokens before the final agent message. Without distillation pressure on the thinking tokens, the agent's output adjusts, but shows oscillatory behavior. When thinking token KL is enabled, the behavior is instead a stable curve and increasing monotonically (Figure 7).

To examine this behavior, we can take a look at the thinking traces for a model trained with and without distillation on the thinking tokens. Without distillation on thinking, the thinking trace for the final message does not mention style at all:

> Okay, I've modified the code to support arrays of state patterns. The original issue was that the user was trying to use multiple conditions with the same key ('active'), which doesn't work in JavaScript objects because duplicate keys aren't allowed.

On the other hand, if distillation is applied to the thinking tokens as well, then the thinking content of the final message includes a range of self-reminders to include specific stylistic components:

> Now I should provide a clear summary to the user about what I did. I should explain the issue, the solution, and ask for their confirmation to proceed or just implement it. I should format this nicely with proper markdown.

6 Discussion and Conclusion

In this work, we introduced a simple targeted guidance approach that combines textual feedback self-distillation with standard on-policy RL. This allows for optimization of rewards criteria that may be difficult or impossible to optimize with policy gradients alone, while remaining stable enough to not interfere with baseline RL performance.

Limitations. One limitation of our proposed framework is that each reward in the suite requires a designed static feedback string. Follow-up work may examine whether strong feedback can be automatically learned, perhaps via a tied RL process. Experimentally, the settings studied in this work focus solely on the multi-turn software engineering setup, and our targets for guidance largely revolve around style and formatting rather than primary intelligence.

Future work. Central to this method is the use of language as communicative medium between the reward and the policy, i.e. the reward judge needs a method for producing targeted textual feedback and the teacher model needs to condition on this information in an effective manner. In this work, we rely on the zero-shot ability of the policy to process this information and human prompting for the judge to produce effective feedback. We anticipate that further in-domain SFT on the policy can improve the effectiveness of the teacher signal, and that potentially reward judges could be fine tuned to produce more targeted hints that minimize reward interference.

The approach in this work focused specifically on a per-token distillation approach to utilizing feedback. However, there may be strong benefits from allowing the teacher to autoregressively generate alternative turns, or otherwise rewrite the on-policy rollouts. Recent work on teacher generation approaches, such as pedagogical RL (Chakraborty et al., 2026), suggest that training teacher models to provide guidance that is closer to the student distribution might be an effective way forward.

Finally, the feedback sources in this work focuses on targeted guidance at the turn-level. This approach fits with the reward criteria that were targeted for this work. A natural question is whether a similar approach could be used for dense guidance on the full rollout. While RL is a strong baseline in this setting, these results indicate that with the right stabilization text feedback may be able to complement with global exploration.

AI use statement

In this work, we used generative AI tools for implementing code, creating plots, locating relevant related work, and to refine and iterate on the writing process. We have not used generative AI tools for research ideation, developing core theories, or designing the structure of the project. We have

reviewed all AI-assisted work and take responsibility for the final content, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International Conference on Machine Learning (ICML)*, 2017.

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations (ICLR)*, 2024.

Anurag Ajay, Aviral Kumar, Pulkit Agrawal, Sergey Levine, and Ofir Nachum. OPAL: Offline primitive discovery for accelerating offline reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2021.

Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, et al. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*, 2021.

Marc G. Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Rémi Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *International Conference on Learning Representations (ICLR)*, 2019.

Souradip Chakraborty, Noah Ziems, Furong Huang, Meng Jiang, Amrit Singh Bedi, and Omar Khattab. Pedagogical rl: Teaching models to teach themselves from privileged information. 2026. https://noahziems.com/pedagogical-rl.

Haolin Chen, Yihao Feng, Zuxin Liu, Weiran Yao, Akshara Prabhakar, Shelby Heinecke, Ricky Ho, Phil Mui, Silvio Savarese, Caiming Xiong, et al. Language models are hidden reasoners: Unlocking latent reasoning capabilities via self-rewarding. *arXiv preprint arXiv:2411.04282*, 2024.

Runpeng Dai, Linfeng Song, Haolin Liu, Zhenwen Liang, Dian Yu, Haitao Mi, Zhaopeng Tu, Rui Liu, Tong Zheng, Hongtu Zhu, and Dong Yu. CDE: Curiosity-driven exploration for efficient reinforcement learning in large language models. *arXiv preprint arXiv:2509.09675*, 2025.

Ken Ding. Hdpo: Hybrid distillation policy optimization via privileged self-distillation. *arXiv preprint arXiv:2603.23871*, 2026.

Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. First return, then explore. *Nature*, 590:580–586, 2021.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations (ICLR)*, 2019.

Kevin Frans, Seohong Park, Pieter Abbeel, and Sergey Levine. Unsupervised zero-shot reinforcement learning via functional reward encodings. In *International Conference on Machine Learning (ICML)*, 2024.

Preston Fu, Kevin Frans, Oleh Rybkin, Sergey Levine, and Aviral Kumar. Long-horizon language model reinforcement learning via progressive point matching. *arXiv preprint arXiv:2609.07303*, 2026.

J. Gao, L. Pan, Y. Wang, R. Zhong, C. Lu, Q. Cai, P. Jiang, and X. Zhao. Navigate the unknown: Enhancing LLM reasoning with intrinsic motivation guided exploration. *arXiv preprint arXiv:2505.17621*, 2025.

Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control. *arXiv preprint arXiv:1611.07507*, 2016.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *International Conference on Learning Representations (ICLR)*, 2024.

Elad Hazan, Sham Kakade, Karan Singh, and Abby Van Soest. Provably efficient maximum entropy exploration. In *International Conference on Machine Learning (ICML)*, 2019.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Woogyeol Jin, Taywon Min, Yongjin Yang, Swanand Ravindra Kadhe, Yi Zhou, Dennis Wei, Nathalie Baracaldo, and Kimin Lee. Entropy-aware on-policy distillation of language models. *arXiv preprint arXiv:2603.07079*, 2026.

Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2002.

Simran Kaur, Narutatsu Ri, Yinghui He, Liam Fowl, and Sanjeev Arora. Rethinking on-policy self-distillation for thinking models. *arXiv preprint arXiv:2607.05184*, 2026.

Jeonghye Kim, Xufang Luo, Minbeom Kim, Sangmook Lee, Dohyung Kim, Jiwon Jeon, Dongsheng Li, and Yuqing Yang. Why does self-distillation (sometimes) degrade the reasoning capability of LLMs? *arXiv preprint arXiv:2603.24472*, 2026.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2016.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D. Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei M. Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal Behbahani, and Aleksandra Faust. Training language models to self-correct via reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2025.

Lisa Lee, Benjamin Eysenbach, Emilio Parisotto, Eric Xing, Sergey Levine, and Ruslan Salakhutdinov. Efficient exploration via state marginal matching. *arXiv preprint arXiv:1906.05274*, 2019.

Gengsheng Li, Tianyu Yang, Junfeng Fang, Mingyang Song, Mao Zheng, Haiyun Guo, Dan Zhang, Jinqiao Wang, and Tat-Seng Chua. Unifying group-relative and self-distillation policy optimization via sample routing. *arXiv preprint arXiv:2604.02288*, 2026.

T. Li, Y. Zhang, P. Yu, S. Saha, D. Khashabi, J. Weston, J. Lanchantin, and T. Wang. Jointly reinforcing diversity and quality in language model generations. *arXiv preprint arXiv:2509.02534*, 2025.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations (ICLR)*, 2024.

Hao Liu and Pieter Abbeel. Behavior from the void: Unsupervised active pre-training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

David Lopez-Paz, Léon Bottou, Bernhard Schölkopf, and Vladimir Vapnik. Unifying distillation and privileged information. In *International Conference on Learning Representations (ICLR)*, 2016.

Kevin Lu and Thinking Machines Lab. On-policy distillation. Thinking Machines Lab: Connectionism, 2025. https://thinkingmachines.ai/blog/on-policy-distillation/.

Zhengxi Lu, Zhiyuan Yao, Zhuowen Han, Zi-Han Wang, Jinyang Wu, Qi Gu, Xunliang Cai, Weiming Lu, Jun Xiao, Yueting Zhuang, et al. Self-distilled agentic reinforcement learning. *arXiv preprint arXiv:2605.15155*, 2026.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Vaskar Nath, Elaine Lau, Anisha Gunjal, Manasi Sharma, Nikhil Baharte, and Sean Hendryx. Adaptive guidance accelerates reinforcement learning of reasoning models. *arXiv preprint arXiv:2506.13923*, 2025.

Seohong Park, Oleh Rybkin, and Sergey Levine. METRA: Scalable unsupervised RL with metric-aware abstraction. In *International Conference on Learning Representations (ICLR)*, 2024.

Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning (ICML)*, 2017.

Emiliano Penaloza, Dheeraj Vattikonda, Nicolas Gontier, Alexandre Lacoste, Laurent Charlin, and Massimo Caccia. Privileged information distillation for language models. *arXiv preprint arXiv:2602.04942*, 2026.

Keqin Peng, Chen Li, Yuanxin Ouyang, Yancheng Yuan, and Liang Ding. Diagnosing and mitigating thinking collapse in on-policy self-distillation. *arXiv preprint arXiv:2607.10805*, 2026.

Karl Pertsch, Youngwoon Lee, and Joseph J. Lim. Accelerating reinforcement learning with learned skill priors. In *Conference on Robot Learning (CoRL)*, 2020.

Alexandre Piché, Ehsan Kamalloo, Rafael Pardinas, Xiaoyin Chen, and Dzmitry Bahdanau. Pipelinerl: Faster on-policy reinforcement learning for long sequence generation. *arXiv preprint arXiv:2509.19128*, 2025.

Yuxiao Qu, Tianjun Zhang, Naman Garg, and Aviral Kumar. Recursive introspection: Teaching language model agents how to self-improve. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Yuxiao Qu, Amrith Setlur, Virginia Smith, Ruslan Salakhutdinov, and Aviral Kumar. POPE: Learning to reason on hard problems via privileged on-policy exploration. *arXiv preprint arXiv:2601.18779*, 2026a.

Yuxiao Qu, Anikait Singh, Yoonho Lee, Amrith Setlur, Ruslan Salakhutdinov, Chelsea Finn, and Aviral Kumar. RLAD: Training LLMs to discover abstractions for solving reasoning problems. In *International Conference on Learning Representations (ICLR)*, 2026b. arXiv:2510.02263.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 627–635, 2011.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning (ICML)*, 2015.

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding progress: Scaling automated process verifiers for LLM reasoning. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2410.08146.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-distillation enables continual learning. *arXiv preprint arXiv:2601.19897*, 2026.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Charlie Snell, Dan Klein, and Ruiqi Zhong. Learning by distilling context. *arXiv preprint arXiv:2209.15189*, 2022.

Yuda Song, Julia Kempe, and Rémi Munos. Outcome-based exploration for LLM reasoning. *arXiv preprint arXiv:2509.06941*, 2025.

Haoran Sun, Yekun Chai, Shuohuan Wang, Yu Sun, Hua Wu, and Haifeng Wang. Curiosity-driven reinforcement learning from human feedback. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2025. arXiv:2501.11463.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. Learning to reason under off-policy guidance. *arXiv preprint arXiv:2504.14945*, 2025.

Tianzhu Ye, Li Dong, Xun Wu, Shaohan Huang, and Furu Wei. On-policy context distillation for language models. *arXiv preprint arXiv:2602.12275*, 2026.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Xuan Zhang, Ruixiao Li, Zhijian Zhou, Long Li, Yulei Qin, Ke Li, Xing Sun, Xiaoyu Tan, Chao Qu, and Yuan Qi. Count counts: Motivating exploration in LLM reasoning with count-based intrinsic rewards. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2510.16614.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

Xiangxin Zhou, Zichen Liu, Anya Sims, Haonan Wang, Tianyu Pang, Chongxuan Li, Liang Wang, Min Lin, and Chao Du. Reinforcing general reasoning without verifiers. In *International Conference on Learning Representations*, volume 2026, pp. 69689–69712, 2026.