

Self-Improving Reasoning Agents via Reinforcement Learning and Iterative Feedback

Ruperto Llamas

Department of Electrical Engineering and Computer Science, University of Missouri,
Columbia, MO, USA.
rupertomail@missouri.edu

Alida Bolander

School of Computing, Clemson University, Clemson, SC, USA.
bolanderalida@clemson.edu

Nino Bosio

School of Information Technology, University of Cincinnati, Cincinnati, OH, USA.
ninowork@uc.edu

Rachana Mahal

Department of Computer Science, University of Houston, Houston, TX, USA.
rachana.work@uh.edu

Abstract

Self-improving reasoning agents represent a class of artificial intelligence systems that refine their own problem-solving behavior through repeated cycles of reinforcement learning, evaluative feedback, and environmental interaction. Unlike static models that are deployed after a single training phase, these agents are designed to treat reasoning as an evolving capability shaped by feedback from users, tools, simulations, and governance mechanisms. This paper examines the system-level architecture, learning dynamics, infrastructure requirements, and policy implications of such agents. It argues that self-improvement is not merely an algorithmic property but a socio-technical process involving data pipelines, reward design, evaluation regimes, deployment constraints, and institutional oversight. The analysis considers reinforcement learning from human preferences, self-play, verbal reinforcement, self-refinement, and trajectory filtering as complementary mechanisms that can support iterative reasoning. It also investigates structural trade-offs among scalability, robustness, fairness, interpretability, and environmental sustainability. Particular attention is given to the risks of reward hacking, feedback loops, distributional shift, and opaque optimization. The paper concludes that durable self-improvement requires layered governance, auditable feedback channels, and infrastructure that can support continuous evaluation without sacrificing safety or equity. The discussion is relevant to researchers and practitioners building reasoning agents for domains such as software engineering, scientific discovery, education, and decision support.

Keywords

self-improving agents; reinforcement learning; iterative feedback; reasoning; governance; infrastructure; robustness; fairness; sustainability; human feedback.

1. Introduction

The emergence of reasoning agents capable of improving their own performance has become a central concern in artificial intelligence research and systems engineering. Early reinforcement learning established that agents can acquire complex behaviors through trial and error, often surpassing hand-coded strategies when the environment provides a sufficiently informative reward signal [1]. More recent work has extended this logic to language-based agents that reason through intermediate steps, call external tools, and revise their outputs in response to feedback [11]. These developments suggest a shift from models as fixed artifacts to agents as adaptive systems that accumulate competence over time. However, the transition from a capable model to a self-improving reasoning agent introduces architectural, infrastructural, and governance challenges that cannot be reduced to model architecture alone. A self-improving agent must decide what to learn, when to update, which feedback to trust, and how to avoid degrading previously acquired capabilities. These are system-level questions involving data provenance, evaluation design, deployment constraints, and institutional accountability [21].

The promise of self-improving reasoning agents lies in their potential to operate in open-ended environments where static training cannot anticipate every task. In software engineering, for example, an agent might propose code, execute tests, observe failures, and revise its strategy across multiple attempts [13]. In scientific discovery, an agent might generate hypotheses, design experiments, and refine its search based on empirical outcomes [14]. In education, an agent might adapt explanations to learner responses and improve its pedagogical tactics over time. In each case, iterative feedback serves as the mechanism through which experience is converted into improved reasoning. Yet the same feedback loops that enable improvement can also amplify bias, reward superficial success, or create brittle policies that fail under distributional shift [18]. The design of self-improving agents therefore requires a careful balance between openness to new information and safeguards against uncontrolled drift.

This paper develops a system-level account of self-improving reasoning agents. It first reviews foundational work on reinforcement learning, language model reasoning, and feedback-based alignment. It then proposes an architectural view that separates reasoning, memory, feedback acquisition, evaluation, and update governance. Subsequent sections analyze reinforcement learning and iterative feedback mechanisms, infrastructure and sustainability concerns, robustness and fairness, and cross-domain evaluation. The final sections consider future directions and conclude with policy-oriented reflections. The central argument is that self-improvement should be understood as a governed process rather than an autonomous endpoint. Agents may improve only insofar as their feedback channels, reward models, evaluation suites, and deployment constraints are designed to support reliable and equitable learning [4]. Without such governance, self-improving systems risk optimizing for measurable proxies while eroding human trust and social value [19].

2. Foundations and Related Work

The intellectual foundations of self-improving reasoning agents draw from reinforcement learning, deep learning, and natural language processing. Reinforcement learning formalizes the problem of an agent interacting with an environment to maximize cumulative reward [1]. Classical methods such as temporal-difference learning and policy gradients provided the basis for agents that learn from experience. Deep learning later enabled reinforcement learning to operate over high-dimensional inputs such as images, speech, and text [8]. The combination of deep neural networks with reinforcement learning produced breakthroughs in

games and control, including self-play systems that mastered complex board games without human demonstrations [7]. These successes demonstrated that iterative self-play and policy improvement can yield superhuman performance in bounded domains. They also revealed that learned policies can be sensitive to reward design, exploration strategies, and the distribution of training environments.

In parallel, transformer-based language models transformed the study of reasoning by providing a general-purpose architecture for sequence modeling and contextual inference [9]. Large-scale pretraining enabled few-shot and zero-shot generalization across many tasks [10]. Chain-of-thought prompting further showed that explicit intermediate reasoning can improve performance on multi-step problems [11]. ReAct-style agents combined reasoning with acting by interleaving thought, tool use, and observation [12]. Reflexion introduced verbal reinforcement, allowing agents to learn from self-generated feedback without updating model weights [13]. Other systems, such as Voyager, demonstrated open-ended skill acquisition through iterative exploration and code generation [14]. Self-Refine and related methods showed that models can improve outputs through iterative critique and revision [15]. STaR and similar bootstrapping approaches demonstrated that reasoning traces can be filtered and reused to train stronger reasoning behavior [16]. These lines of work collectively establish that reasoning can be treated as a learnable and revisable process rather than a fixed inference pattern.

Alignment research has contributed complementary insights. Reinforcement learning from human feedback demonstrated that human preferences can be used to shape model behavior [3]. Instruction tuning with human feedback further improved helpfulness and harmlessness in large language models [4]. Related work on helpful and harmless assistants explored the tension between capability and safety [5]. Proximal policy optimization provided a stable optimization method that became widely used in feedback-based training [6]. At the same time, concerns about AI safety highlighted challenges such as reward hacking, negative side effects, and scalable oversight [18]. Critical scholarship on large-scale language models warned about environmental costs, bias, and concentration of power [19]. Model cards and documentation practices emerged as mechanisms for transparency and accountability [20]. Research on hidden technical debt in machine learning systems showed that deployment and maintenance costs often exceed initial model development [21]. Energy and policy analyses quantified the carbon footprint of deep learning and called for greater efficiency and transparency [22]. These works frame self-improvement not only as a technical capability but also as an infrastructural and governance problem.

3. Architecture of Self-Improving Reasoning Agents

A self-improving reasoning agent can be understood as a layered system composed of reasoning, memory, feedback acquisition, evaluation, and update governance. The reasoning layer generates candidate solutions, plans, explanations, or actions. It may use chain-of-thought prompting, tool invocation, or structured decomposition to handle complex tasks [11]. The memory layer stores past interactions, successful strategies, failed attempts, and environmental observations. Memory can be short-term, such as a context window, or long-term, such as a vector database or skill library [14]. The feedback acquisition layer gathers signals from users, automated evaluators, execution environments, or other agents. Feedback may be explicit, such as a rating, or implicit, such as whether code compiles or a simulation succeeds [13]. The evaluation layer determines whether feedback is reliable, whether the agent has improved, and whether updates should be accepted. The update governance layer

enforces constraints on learning, such as safety filters, rollback rules, and human approval requirements. This layered view clarifies that self-improvement is not a single algorithm but a pipeline of interdependent components.

The separation between reasoning and update governance is especially important. If an agent can modify its own policy without oversight, it may exploit loopholes in the reward function or overfit to spurious feedback [18]. If governance is too restrictive, the agent may fail to learn from novel situations. A practical architecture often combines offline training, online adaptation, and periodic consolidation. Offline training uses curated datasets and human feedback to establish baseline capabilities [4]. Online adaptation allows the agent to adjust to specific tasks or environments, sometimes through in-context learning rather than weight updates [12]. Periodic consolidation selectively incorporates validated improvements into the base model or a retrieval memory. This hybrid approach balances flexibility with stability. It also creates an audit trail that can be inspected after failures. The architectural trade-off is between plasticity, which enables improvement, and stability, which preserves reliable behavior.

Another architectural consideration is modularity. A monolithic agent that learns end-to-end may be difficult to debug and govern. A modular design separates perception, reasoning, planning, tool use, and evaluation into distinct components with defined interfaces. This modularity can improve interpretability because failures can be localized to specific modules. It can also support incremental improvement, allowing one module to be updated without destabilizing the entire system. However, modularity introduces coordination costs and interface risks. If modules optimize different objectives, the overall system may behave inconsistently. A self-improving agent therefore requires an integration layer that aligns module-level incentives with system-level goals. This integration layer may use shared representations, common reward models, or negotiated constraints. The design challenge resembles that of large socio-technical infrastructures, where local optimization must be reconciled with global reliability [21].

4. Reinforcement Learning and Iterative Feedback

Reinforcement learning provides the primary machinery for self-improvement because it formalizes learning from interaction. An agent receives observations, takes actions, and receives rewards that indicate the desirability of outcomes [1]. In reasoning agents, actions may include generating a step, calling a tool, asking a clarifying question, or revising a prior answer. Rewards may come from human preferences, automated verifiers, task success, or self-evaluation [3]. Policy gradient methods and actor-critic methods update the agent's policy to increase expected reward [6]. Proximal policy optimization is widely used because it constrains updates to avoid destructive policy changes [6]. However, reinforcement learning in open-ended reasoning domains faces challenges such as sparse rewards, delayed feedback, and non-stationary environments. These challenges are compounded when the agent's actions affect the feedback it receives, creating feedback loops that can reinforce bias or deception [18].

Iterative feedback mechanisms extend reinforcement learning by incorporating multiple rounds of critique and revision. Self-Refine allows a model to generate an initial output, critique it, and produce an improved version without weight updates [15]. Reflexion converts feedback into verbal self-reflections that are stored in memory and used in later attempts [13]. STaR filters reasoning traces and fine-tunes on those that lead to correct answers, effectively bootstrapping reasoning from reasoning [16]. ReAct interleaves reasoning and acting,

enabling the agent to gather external information before committing to an answer [12]. These methods show that improvement can occur at different timescales: within a single episode, across episodes through memory, and across training cycles through parameter updates. A robust self-improving agent may combine all three. Within-episode refinement handles immediate errors, memory-based reflection supports rapid adaptation, and parameter updates consolidate long-term gains. The integration of these timescales requires careful scheduling so that short-term corrections do not conflict with long-term learning.

Recent work on trajectory filtering and alignment for small-parameter language models illustrates how reinforcement learning can be made more scalable when reasoning paths are pruned before optimization [17]. This approach is relevant to system-level design because it suggests that not all generated reasoning traces should be treated equally. Filtering can reduce computational cost, improve sample efficiency, and limit exposure to low-quality data. However, filtering also introduces selection bias. If the filter rewards only concise or superficially correct traces, the agent may learn to suppress uncertainty, omit important caveats, or imitate brittle reasoning patterns. The design of filters therefore has governance implications. Filtering criteria should be documented, audited, and aligned with domain-specific norms. In safety-critical domains, for example, a trace that reaches the correct answer through invalid reasoning should not be reinforced. The same logic applies to reward models: they are not neutral arbiters of quality but encoded value judgments that shape future behavior [4].

5. Infrastructure, Deployment, and Sustainability

Self-improving reasoning agents depend on substantial infrastructure. Training requires compute, storage, networking, and data pipelines. Inference requires serving infrastructure that can handle variable workloads, tool calls, and memory retrieval. Continuous improvement requires logging, monitoring, evaluation, and rollback systems. These infrastructural elements are often underestimated in research prototypes but become critical in production [21]. A deployed agent must operate under latency constraints, cost budgets, privacy requirements, and reliability expectations. It must also handle distributional shift as user behavior, data sources, and environments change. Infrastructure that supports self-improvement must therefore be observable and controllable. Operators need to know which feedback was used, how updates were validated, and what changed in the agent’s behavior. Without such observability, self-improvement becomes a source of technical debt rather than a durable capability.

Sustainability is a related concern. Large-scale language models and reinforcement learning pipelines consume significant energy and resources [22]. Iterative self-improvement can multiply these costs if every improvement cycle requires extensive retraining or large-scale human annotation. A sustainable architecture should prioritize sample efficiency, reuse of validated experience, and targeted updates. Parameter-efficient fine-tuning, retrieval-based memory, and modular adaptation can reduce the need for full retraining. However, efficiency gains may be offset by increased inference costs if agents use longer reasoning chains or call multiple tools. The system-level trade-off is between training efficiency and inference efficiency. A model that is cheap to train but expensive to run may be unsustainable at scale. Conversely, a model that is expensive to train but efficient at inference may be justified for high-value applications. These decisions should be informed by lifecycle assessments rather than narrow benchmarks [22]. Sustainability also includes human sustainability: annotation teams, moderators, and operators must not be subjected to exploitative labor conditions [19].

Deployment introduces additional constraints. A self-improving agent may be deployed in a regulated environment where changes to its behavior require approval. In healthcare, finance, or public administration, autonomous updates may be legally or ethically unacceptable without human review. In consumer applications, users may expect consistent behavior and may not consent to being part of an ongoing training loop. These constraints suggest a spectrum of deployment models. At one end, the agent may be frozen after training and updated only through controlled releases. At the other end, the agent may adapt continuously within bounded parameters. Most practical systems will occupy a middle ground, with limited online adaptation and periodic consolidation. The choice depends on risk tolerance, regulatory context, and the availability of evaluation infrastructure. Governance must specify who is accountable when an adaptive agent causes harm, especially when the harmful behavior emerged from feedback rather than explicit programming [20].

6. Robustness, Fairness, and Governance

Robustness in self-improving reasoning agents refers to the ability to maintain reliable performance under distributional shift, adversarial inputs, and noisy feedback. An agent that improves on a narrow distribution may degrade on out-of-distribution tasks. It may also become vulnerable to reward hacking, where it discovers unintended ways to maximize reward without achieving the intended goal [18]. Robustness requires evaluation beyond average accuracy. It requires stress tests, adversarial probes, and counterfactual analysis. It also requires monitoring after deployment, because self-improvement can introduce new failure modes over time. A robust agent should be able to detect when it is uncertain, request help, or fall back to a conservative policy. These capabilities are not merely technical features; they are governance mechanisms that allocate authority between the agent and human overseers.

Fairness is equally important. Feedback data often reflects historical inequities, and reward models trained on such data can perpetuate discrimination [19]. If a self-improving agent learns from user ratings, it may optimize for the preferences of the most vocal or powerful users. If it learns from automated metrics, it may ignore marginalized groups whose needs are not represented in the metrics. Fairness therefore cannot be an afterthought. It must be embedded in feedback collection, reward design, evaluation, and update approval. This may involve stratified evaluation, participatory design, and explicit constraints on disparate impact [20]. It may also require transparency about the agent's limitations. Model cards and data statements are useful but insufficient if they are not tied to enforceable accountability. Fairness in self-improving systems is dynamic because the agent's behavior changes over time. Continuous monitoring and community input are necessary to detect emerging harms.

Governance frameworks for self-improving agents must address several layers. At the technical layer, they specify update rules, safety constraints, and rollback procedures. At the organizational layer, they assign responsibility for data collection, model evaluation, and incident response. At the regulatory layer, they define legal obligations for transparency, liability, and redress. At the societal layer, they facilitate public deliberation about acceptable uses and limits. These layers interact. A technical safety constraint may be ineffective if organizational incentives reward rapid deployment over caution. A regulatory requirement may be undermined if technical systems cannot produce auditable records. Effective governance therefore requires alignment across layers [21]. It also requires international coordination, because agents can be deployed across borders and trained on globally sourced

data. The challenge resembles environmental and financial regulation, where local actions have global consequences and enforcement is uneven.

7. Evaluation, Case Illustrations, and Cross-Domain Comparisons

Evaluation of self-improving reasoning agents is difficult because the target is moving. Static benchmarks measure performance at a point in time, but they do not capture whether the agent is improving in desirable ways. An agent may score higher on a benchmark while becoming less robust, less fair, or less interpretable. A comprehensive evaluation framework should include task performance, sample efficiency, robustness, fairness, calibration, and human trust. It should also assess the learning process itself. Which feedback was used? How were updates validated? Did the agent become more or less reliant on shortcuts? Did it develop new capabilities or merely memorize patterns? Process-oriented evaluation is more expensive than outcome-oriented evaluation, but it is necessary for high-stakes domains [20].

Case illustrations help clarify these issues. In software engineering, a self-improving coding agent might generate unit tests, run them, and revise code until tests pass. This loop can improve productivity, but it can also reinforce code that passes tests while violating security or maintainability standards [13]. In scientific discovery, an agent might propose experiments and refine hypotheses based on results. This can accelerate research, but it can also amplify publication bias if it favors hypotheses that yield positive results [14]. In education, an agent might adapt to student responses and improve its explanations. This can personalize learning, but it can also narrow the curriculum if feedback favors easy-to-measure outcomes. In customer service, an agent might optimize for resolution speed, leading to superficial answers that frustrate users. These examples show that domain-specific values must shape reward design and evaluation. There is no universal reward function for reasoning quality [4].

Cross-domain comparison reveals common structural patterns. Self-improving agents tend to concentrate power in the entities that control compute, data, and feedback channels. They tend to create feedback loops that advantage early successes and disadvantage alternative approaches. They tend to obscure accountability because improvements emerge from complex interactions rather than explicit instructions. They tend to require continuous investment in evaluation and monitoring, which favors large organizations. These patterns suggest that self-improvement is not inherently democratizing. Without deliberate design, it may reproduce existing inequalities [19]. On the other hand, modular architectures, open evaluation, and participatory governance can distribute benefits more broadly. The outcome depends on institutional choices as much as technical ones.

8. Future Directions

Future research should develop architectures that make self-improvement auditable and reversible. This includes logging feedback provenance, tracking policy changes, and providing operators with tools to simulate the effects of updates before deployment. It also includes methods for uncertainty-aware learning, where agents recognize when feedback is unreliable and defer to human judgment. Another direction is multi-agent self-improvement, where agents critique and teach one another. Multi-agent systems can accelerate learning, but they can also create echo chambers and collusive behavior. Governance mechanisms for multi-agent improvement are still immature. Research should explore how to design incentives that reward honesty, cooperation, and robustness rather than narrow task success [18].

Another promising direction is the integration of symbolic reasoning, retrieval, and neural learning. Hybrid systems may combine the flexibility of neural models with the verifiability of symbolic methods. They may also support more structured feedback, such as proofs, constraints, or formal specifications. However, hybrid systems introduce new complexity and may be difficult to scale. Research should investigate when hybrid approaches outperform purely neural ones and how they can be governed. Sustainability should also be a first-class design goal. This includes energy-efficient training, carbon-aware scheduling, and lifecycle accounting [22]. Finally, policy research should focus on standards for self-improving systems, including transparency, liability, and redress. International coordination will be necessary to prevent a race to the bottom in safety and fairness [20].

9. Conclusion

Self-improving reasoning agents represent a significant evolution in artificial intelligence, moving from static models to adaptive systems that refine their behavior through reinforcement learning and iterative feedback. This paper has argued that self-improvement is a socio-technical process shaped by architecture, infrastructure, governance, and institutional context. Reinforcement learning provides powerful mechanisms for improvement, but it also introduces risks such as reward hacking, bias amplification, and brittle optimization. Iterative feedback methods such as self-refinement, verbal reinforcement, and trajectory filtering can enhance reasoning, but they require careful design to avoid reinforcing shortcuts and suppressing uncertainty. Infrastructure for deployment, monitoring, and sustainability is essential for durable improvement. Fairness and robustness cannot be guaranteed by technical fixes alone; they require continuous evaluation, participatory governance, and enforceable accountability.

The central lesson is that self-improving agents should be designed as governed systems rather than autonomous optimizers. Governance must operate across technical, organizational, regulatory, and societal layers. It must be capable of learning as well, adapting to new evidence and emerging harms. The alternative is a proliferation of opaque systems that optimize measurable proxies while eroding trust and equity. With deliberate design, self-improving reasoning agents can augment human capability in domains such as science, engineering, education, and public administration. Without such design, they risk becoming engines of unintended consequences. The path forward requires interdisciplinary collaboration among researchers, engineers, policymakers, and communities. The future of self-improving reasoning is not predetermined by algorithms; it will be shaped by the choices made in building and governing these systems.

References

1. Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction (2nd ed.). MIT Press.
2. Russell, S. J., & Norvig, P. (2021). Artificial intelligence: A modern approach (4th ed.). Pearson.
3. Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., & Amodei, D. (2017). Deep reinforcement learning from human preferences. *Advances in Neural Information Processing Systems*, 30, 4299–4307.
4. Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens,

- M., Askell, A., Welinder, P., Christiano, P., Leike, J., & Lowe, R. (2022). Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35, 27730–27744.
5. Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., DasSarma, N., Drain, D., Fort, H., Ganguli, D., Henighan, T., Joseph, N., Kadavath, S., Kernion, J., Conerly, T., El-Showk, S., Elhage, N., Hatfield-Dodds, Z., Hernandez, D., Johnston, S., ... Kaplan, J. (2022). Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv*. <https://doi.org/10.48550/arXiv.2204.05862>
 6. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv*. <https://doi.org/10.48550/arXiv.1707.06347>
 7. Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., & Hassabis, D. (2018). A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419), 1140–1144.
 8. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444.
 9. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 5998–6008.
 10. Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., ... Amodei, D. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 1877–1901.
 11. Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35, 24824–24837.
 12. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). ReAct: Synergizing reasoning and acting in language models. *International Conference on Learning Representations*.
 13. Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., & Yao, S. (2023). Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 8634–8652.
 14. Wang, G., Xie, Y., Jiang, Y., Mandlekar, A., Xiao, C., Zhu, Y., Fan, L., & Anandkumar, A. (2023). Voyager: An open-ended embodied agent with large language models. *arXiv*. <https://doi.org/10.48550/arXiv.2305.16291>
 15. Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegreffe, S., Alon, U., Dziri, N., Prabhume, S., Yang, Y., Gupta, S., Majumder, B. P., Hermann, K., Welleck, S., Yazdanbakhsh, A., & Clark, P. (2023). Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36, 46534–46594.
 16. Zelikman, E., Wu, Y., Mu, J., & Goodman, N. D. (2022). STaR: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35, 15476–15488.

17. Li, Q. (2026, May). TrajLite PPO: Scalable Reasoning Path Filtering and Alignment for Small Parameter Large Language Models. In 2026 2nd International Conference on Artificial Intelligence and Digital Ethics (ICAIDE) (pp. 29-32). IEEE.
18. Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., & Mané, D. (2016). Concrete problems in AI safety. arXiv. <https://doi.org/10.48550/arXiv.1606.06565>
19. Bender, E. M., Gebru, T., McMillan-Major, A., & Shmitchell, S. (2021). On the dangers of stochastic parrots: Can language models be too big? Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency, 610–623. <https://doi.org/10.1145/3442188.3445922>
20. Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., Spitzer, E., Raji, I. D., & Gebru, T. (2019). Model cards for model reporting. Proceedings of the Conference on Fairness, Accountability, and Transparency, 220–229. <https://doi.org/10.1145/3287560.3287596>
21. Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., & Dennison, D. (2015). Hidden technical debt in machine learning systems. Advances in Neural Information Processing Systems, 28, 2503–2511.
22. Strubell, E., Ganesh, A., & McCallum, A. (2019). Energy and policy considerations for deep learning in NLP. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 3645–3650. <https://doi.org/10.18653/v1/P19-1355>