

Hierarchical Reinforcement Learning for Tool-Augmented Reasoning in Large Language Models

Peyton Christiansen

Department of Electrical Engineering and Computer Science, University of Kansas, Lawrence,
KS, USA.

peytonmail@ku.edu

Tobiasz Latek

School of Electrical Engineering and Computer Science, Oregon State University, Corvallis,
OR, USA.

latektobiasz@oregonstate.edu

Gaspar Folch

Department of Computer Science, George Mason University, Fairfax, VA, USA.

folchgaspar@gmu.edu

Abstract

Large language models have become general-purpose substrates for reasoning, yet their capacity to act reliably in open environments depends increasingly on tool invocation, memory management, and multi-step orchestration. This paper examines hierarchical reinforcement learning as an organizing architecture for tool-augmented reasoning in large language models. Rather than treating tool use as a single flat action, the proposed perspective separates strategic goal selection from tactical tool execution and from reflective verification. This separation aligns with temporal abstraction in reinforcement learning while respecting the semantic flexibility of language models. The paper discusses system-level trade-offs involving latency, cost, reliability, safety, fairness, sustainability, and governance. It argues that hierarchical reinforcement learning can improve long-horizon task decomposition, credit assignment, and error recovery, but only when coupled with robust infrastructure, observability, permissioned tool access, and continuous evaluation. The analysis also considers deployment in cloud, edge, and hybrid environments, where tool ecosystems introduce technical debt and accountability gaps. Policy implications include auditability, contestability, data minimization, and equitable access to computational resources. The paper concludes that hierarchical reinforcement learning is not merely an algorithmic refinement but a socio-technical design pattern whose benefits depend on institutional safeguards, transparent monitoring, and lifecycle governance. The discussion synthesizes foundational work on transformers, instruction tuning, tool learning, hierarchical control, safety, and operational machine learning systems.

Keywords

hierarchical reinforcement learning, tool-augmented reasoning, large language models, system architecture, governance, robustness, fairness, sustainability.

1. Introduction

The emergence of transformer-based language models has reshaped the landscape of artificial intelligence by enabling broad transfer across tasks without task-specific retraining [1].

Scaling these models has produced few-shot and in-context learning capabilities that resemble general-purpose reasoning, although the mechanisms remain statistical and sensitive to prompt design [2]. Instruction tuning and preference-based alignment have further improved the usability of these models in interactive settings [3]. Despite these advances, large language models remain limited when tasks require precise external actions, fresh information, or verification against structured environments. Tool-augmented reasoning addresses this limitation by allowing models to call search engines, calculators, databases, application programming interfaces, and domain-specific solvers [4]. Toolformer-style approaches demonstrate that models can learn when and how to invoke tools, but the resulting behavior is often myopic and brittle [5]. Tool learning with foundation models extends this agenda into planning, selection, and execution, yet it also exposes unresolved questions about control, safety, and operational reliability [6].

Hierarchical reinforcement learning offers a conceptual vocabulary for structuring such systems. In classical formulations, temporal abstraction allows an agent to operate at multiple levels of decision-making, where high-level policies select subgoals and low-level policies execute actions over extended horizons [7]. Feudal networks and related architectures illustrate how learned hierarchies can decompose complex tasks into manageable modules [8]. Data-efficient hierarchical reinforcement learning further demonstrates that abstraction can improve sample efficiency and transfer in domains where flat policies struggle [9]. These ideas are relevant to tool-augmented language models because tool use is rarely a single atomic act. A user request may require clarification, information retrieval, computation, synthesis, validation, and revision. Treating all of these as one flat action space creates credit assignment problems, unstable exploration, and limited interpretability.

This paper argues that hierarchical reinforcement learning for tool-augmented reasoning should be understood as a system architecture rather than a single optimization technique. The central claim is that hierarchical control can improve long-horizon reasoning when it is paired with explicit tool governance, observability, and lifecycle management. The analysis proceeds from conceptual foundations to architecture, deployment, robustness, fairness, sustainability, policy, case illustrations, evaluation, and governance. Throughout, the paper emphasizes structural trade-offs. Hierarchies can reduce search complexity but introduce coordination overhead. Tool access can increase factual grounding but create security and privacy risks. Reinforcement learning can optimize behavior but amplify reward hacking and distributional shift. The appropriate response is not simply better algorithms but a socio-technical design that aligns incentives, infrastructure, and accountability.

2. Conceptual Foundations of Tool-Augmented Reasoning

Tool-augmented reasoning can be understood as the externalization of cognitive labor. Language models are strong at pattern completion and semantic association, but they are weak at exact arithmetic, real-time retrieval, and stateful interaction. Tools compensate for these weaknesses by providing specialized functions with clearer input-output contracts [4]. In ReAct-style systems, reasoning traces and actions are interleaved, allowing the model to alternate between deliberation and environmental feedback [5]. In tool-learning frameworks, the model must decide which tool to use, when to use it, and how to interpret the result [6]. These decisions form a policy problem, even when the underlying model is not explicitly trained with reinforcement learning. The presence of sequential decision-making, delayed outcomes, and uncertain environments makes reinforcement learning a natural formal lens.

Hierarchical reinforcement learning addresses the problem of long-horizon decision-making by introducing abstraction over time and action. The options framework formalizes temporally extended actions that can be initiated, executed, and terminated under learned or predetermined conditions [7]. Feudal networks separate a manager, which sets abstract goals, from workers, which pursue those goals in the environment [8]. Data-efficient hierarchical methods show that off-policy learning and goal relabeling can make abstraction practical in complex domains [9]. For language-based agents, the hierarchy can be expressed semantically. A high-level policy might decide to gather information, perform computation, or request human confirmation. A mid-level policy might select a database, search engine, or code interpreter. A low-level policy might generate the exact query, parse the response, and decide whether the result is sufficient.

The partially observable nature of tool-augmented reasoning strengthens the case for hierarchy. In partially observable stochastic domains, agents must maintain beliefs about hidden state and act under uncertainty [10]. Language models often lack direct access to the state of external systems, including authentication status, database freshness, and API rate limits. A hierarchical controller can separate epistemic actions, such as checking a source, from pragmatic actions, such as committing a transaction. This separation can improve robustness because verification becomes an explicit subgoal rather than an afterthought. It can also improve safety because high-risk actions can be gated by higher-level policies that consider permissions, reversibility, and user intent.

However, hierarchy also introduces new failure modes. A high-level policy may select an inappropriate subgoal, and a low-level policy may execute it competently but harmfully. Misalignment between levels can produce behavior that appears coherent locally but is globally undesirable. Safety research has long warned about specification gaming, side effects, and scalable oversight [11]. Human-compatible design emphasizes that control must be retained through mechanisms such as interruptibility, transparency, and value alignment [12]. In tool-augmented systems, these concerns are concrete. A model that can send emails, modify files, or execute code must be governed by permissions, confirmation thresholds, and audit trails. Hierarchical reinforcement learning can support such governance, but it cannot replace it.

3. Hierarchical Reinforcement Learning as an Organizing Architecture

A hierarchical architecture for tool-augmented reasoning typically contains three or more levels. At the highest level, a strategic policy interprets user intent, identifies constraints, and selects a mode of operation. At the middle level, a tactical policy decomposes the chosen mode into tool-specific subgoals. At the lowest level, an execution policy generates tool calls, parses outputs, and decides whether the subgoal has been satisfied. This structure mirrors options and feudal hierarchies [7][8], but it is adapted to the semantic and compositional nature of language. The strategic level may operate on natural language summaries, the tactical level on structured tool schemas, and the execution level on formatted inputs and outputs. The separation allows each level to be trained or prompted with different objectives and different granularities of feedback.

Temporal abstraction offers several benefits. First, it reduces the effective horizon of decision-making because high-level policies choose among temporally extended subgoals rather than primitive actions. Second, it improves credit assignment because rewards can be associated with subgoal completion rather than only final outcomes. Third, it supports transfer because subgoals such as retrieval, calculation, and verification can be reused across tasks.

Fourth, it enables modular governance because permissions and safety checks can be attached to subgoal categories. Empirical work in deep reinforcement learning shows that hierarchical and modular structures can improve exploration and stability in complex environments [18]. Similar benefits are plausible in tool-augmented language models, especially when tasks require dozens of steps and multiple external dependencies.

The integration of hierarchical reinforcement learning with large language models raises distinctive design questions. One question concerns whether the hierarchy should be learned end-to-end, hand-designed, or hybrid. End-to-end learning offers adaptability but requires large amounts of interaction data and stable reward signals. Hand-designed hierarchies are interpretable but may fail to generalize. Hybrid approaches use language models as high-level planners and reinforcement learning as a lower-level controller for tool selection and execution. Recent work on scalable reasoning path filtering and alignment for smaller parameter models suggests that carefully curated trajectories and preference alignment can improve reasoning without requiring the largest possible models [17]. This finding is relevant because hierarchical systems often need many small decisions; efficient filtering and alignment can reduce the cost of training and deployment.

Another question concerns multi-agent decomposition. A tool-augmented system may involve several language models or specialized agents, each responsible for a subdomain. One agent might retrieve documents, another might write code, and a third might critique the result. Hierarchical reinforcement learning can coordinate these agents through shared subgoals and centralized or decentralized control. However, multi-agent systems introduce communication overhead, credit assignment ambiguity, and emergent behavior that is difficult to audit. The design must therefore include explicit roles, communication protocols, and fallback procedures. Hierarchy can provide structure, but it can also create opacity if responsibilities are not documented and monitored.

4. System Architecture and Infrastructure for Deployment

From a system perspective, hierarchical reinforcement learning for tool-augmented reasoning requires an orchestration layer that manages state, tools, policies, and memory. The orchestration layer maintains a task graph, tracks subgoal progress, routes requests to appropriate tools, and enforces constraints. A tool registry provides metadata about each tool, including input schema, output schema, latency profile, cost, reliability, and permission requirements. A policy manager selects among high-level strategies and mid-level tool plans. A memory subsystem stores conversation state, retrieved evidence, intermediate results, and audit logs. These components must be integrated with identity management, access control, and observability infrastructure. Without such integration, hierarchical policies may call tools without authorization, repeat failed actions, or leak sensitive information.

Deployment trade-offs are substantial. Cloud deployment offers elasticity, centralized monitoring, and access to powerful models, but it introduces latency, data transfer costs, and privacy concerns. Edge deployment reduces latency and can keep sensitive data local, but it limits model size and tool availability. Hybrid deployment attempts to balance these concerns by placing latency-sensitive or privacy-sensitive components at the edge and compute-intensive components in the cloud. Hierarchical reinforcement learning can support hybrid deployment by assigning different levels of the hierarchy to different environments. For example, the execution level might run locally, while the strategic level runs in the cloud. However, this division requires careful synchronization and fallback logic. If connectivity fails, the local system must degrade gracefully rather than execute stale or unsafe plans.

Operational machine learning systems are prone to hidden technical debt, including entanglement, undeclared consumers, and configuration drift [16]. Tool-augmented reasoning amplifies these risks because each tool is an external dependency with its own versioning, failure modes, and data contracts. A hierarchical policy that worked in development may fail in production when a tool changes its response format or rate limits. Continuous integration and continuous deployment must therefore include contract testing, shadow deployment, canary releases, and rollback mechanisms. Observability must capture not only final outputs but also intermediate subgoals, tool calls, latencies, and errors. This trace data is essential for debugging, auditing, and improving the hierarchy over time.

Governance and infrastructure are inseparable. Permissioned tool access should be based on least privilege, with separate credentials for read-only and write operations. High-impact actions, such as financial transactions or medical record updates, should require multi-factor approval or human confirmation. Data minimization should limit the information sent to external tools. Audit logs should record who or what initiated an action, which subgoal was active, which tool was called, and what result was returned. These measures are not merely compliance overhead; they are functional requirements for trustworthy hierarchical agents. They also provide the feedback signals needed for reinforcement learning, because outcomes can be attributed to specific subgoals and tool interactions.

5. Robustness, Safety, and Governance

Robustness in tool-augmented reasoning means maintaining acceptable performance under distributional shift, adversarial inputs, tool failures, and ambiguous user intent. Hierarchical policies can improve robustness by introducing verification subgoals and fallback paths. If a retrieval tool returns low-quality evidence, a high-level policy can request another source. If a code execution tool fails, a mid-level policy can attempt debugging or choose an alternative interpreter. If a user request is underspecified, the strategic level can ask clarifying questions rather than guessing. These behaviors resemble metacognitive control. However, they depend on reliable uncertainty estimation and on the ability to recognize failure. Language models are often overconfident, and reinforcement learning can exacerbate this if rewards favor confident answers over calibrated ones.

Safety concerns are particularly acute when tools have real-world effects. Research on concrete problems in artificial intelligence identifies reward hacking, negative side effects, and scalable oversight as core challenges [11]. Human-compatible approaches emphasize corrigibility, transparency, and value alignment [12]. In hierarchical tool use, safety must be enforced at multiple levels. The strategic level should refuse or escalate harmful goals. The tactical level should constrain tool selection according to policy. The execution level should validate inputs and outputs against safety filters. These layers should not rely solely on the language model’s internal judgment. They should include external guardrails, such as content filters, permission checks, rate limits, and human-in-the-loop review for high-risk actions.

Governance of tool access is a socio-technical problem. Who decides which tools are available to which agents? How are permissions granted, revoked, and audited? How are conflicts between user intent, organizational policy, and legal requirements resolved? These questions cannot be answered by reinforcement learning alone. They require institutional design, clear accountability, and mechanisms for contestation. Ethical and social risks of language models include misinformation, bias, privacy violations, and exclusion [15]. Foundation models introduce additional risks related to concentration of power, environmental impact, and labor displacement [14]. A hierarchical agent that can invoke

many tools may amplify these risks because it can act across domains and at scale. Governance must therefore be proactive rather than reactive, with impact assessments, red-teaming, and independent audits.

Monitoring and audit are essential for robustness and governance. A hierarchical reinforcement learning system should log the hierarchy of decisions, not just the final output. This includes the selected strategy, the decomposed subgoals, the tools invoked, the parameters passed, the responses received, and the final answer. Such logs enable incident investigation, bias detection, and performance tuning. They also support accountability: if an agent causes harm, investigators can determine whether the failure was due to a model error, a tool error, a policy gap, or an institutional oversight. However, logging must be balanced against privacy and data protection. Logs may contain sensitive user data, proprietary information, or personal identifiers. Retention policies, encryption, and access controls are necessary to prevent logging from becoming a new source of risk.

6. Fairness, Sustainability, and Policy Implications

Fairness in tool-augmented reasoning has both distributive and procedural dimensions. Distributive fairness concerns who has access to capable models, reliable tools, and computational resources. Procedural fairness concerns how decisions are made, whether affected parties can contest them, and whether outcomes are consistent across groups. Large language models can reproduce and amplify social biases present in training data [13]. When these models are coupled with tools, bias can be compounded. A hiring assistant that retrieves candidate information from biased sources may produce discriminatory rankings. A credit assistant that calls a scoring API may inherit historical inequities. Hierarchical reinforcement learning can optimize for fairness objectives, but fairness is not a single scalar. It involves trade-offs among demographic parity, equal opportunity, calibration, and individual fairness. These trade-offs require democratic deliberation, not only technical optimization.

Sustainability is another system-level concern. Training and deploying large models require substantial energy and water for cooling, as well as rare earth materials and specialized hardware [13][14]. Tool-augmented reasoning can increase energy consumption because each tool call may involve additional computation, network transmission, and data storage. Hierarchical reinforcement learning can mitigate this by selecting cheaper or more efficient tools when appropriate. It can also schedule non-urgent tasks during periods of low carbon intensity. However, these benefits depend on accurate cost models and on incentives that reward efficiency rather than maximum capability. Without such incentives, hierarchical agents may default to expensive tools and large models even when simpler alternatives suffice.

Policy implications extend beyond technical efficiency. Regulators may require transparency about automated decision-making, especially in domains such as healthcare, finance, employment, and criminal justice. Hierarchical reinforcement learning can support transparency by making subgoals and tool calls explicit, but it can also obscure accountability if responsibility is distributed across levels. Legal frameworks must clarify who is liable when an autonomous agent causes harm: the user, the developer, the tool provider, or the organization deploying the system. Data protection laws impose constraints on cross-border data flows and on automated processing of personal data. Sector-specific regulations add further requirements, such as clinical validation for medical tools or fiduciary duties for financial advice. A hierarchical architecture should be designed with these requirements in mind, not retrofitted after deployment.

7. Case Illustrations and Cross-Domain Comparisons

In healthcare, a tool-augmented language model might assist with clinical documentation, literature retrieval, and diagnostic support. A hierarchical policy could separate information gathering from differential diagnosis and from treatment recommendation. The strategic level would ensure that recommendations are framed as decision support rather than autonomous medical advice. The tactical level would select trusted medical databases and guidelines. The execution level would extract relevant evidence and flag uncertainty. Reinforcement learning could optimize for accuracy, safety, and clinician satisfaction, but reward design is delicate. Overemphasizing speed could increase diagnostic errors, while overemphasizing caution could reduce usefulness. Governance would require audit trails, privacy protections, and regulatory compliance.

In finance, hierarchical tool use might involve market data retrieval, risk calculation, compliance checking, and transaction execution. The high-level policy must balance return objectives against regulatory constraints and fiduciary duties. The tactical level must select authorized data sources and execution venues. The execution level must validate trades and detect anomalies. Reinforcement learning can improve execution efficiency, but financial markets are adversarial and non-stationary. A policy that performs well in backtesting may fail in live markets due to regime shifts or manipulation. Robustness requires stress testing, circuit breakers, and human oversight. The cost of failure is high, so governance must be conservative.

In robotics and embodied systems, hierarchical reinforcement learning has a longer history. Deep reinforcement learning has achieved human-level control in simulated and physical domains [18], and self-play methods have mastered complex games [19]. Language models can provide high-level semantic goals for robots, while hierarchical controllers handle low-level motor actions. This division is natural because language is better at abstraction and planning, while control policies are better at continuous actuation. The main challenges are grounding, safety, and sim-to-real transfer. A robot that misinterprets a language command can cause physical harm. Verification, constraints, and emergency stop mechanisms are essential.

Cross-domain comparisons reveal recurring patterns. In all domains, hierarchy helps manage long horizons but introduces coordination costs. Tool access improves capability but expands the attack surface. Reinforcement learning enables adaptation but requires stable rewards and safe exploration. Governance is necessary but difficult to standardize across sectors. The most successful deployments are likely to be those that treat the system as a socio-technical infrastructure rather than a standalone model. This means investing in data pipelines, identity management, monitoring, incident response, and stakeholder engagement. It also means accepting that some tasks should not be fully automated, especially when stakes are high and accountability is diffuse.

8. Evaluation, Monitoring, and Operational Lifecycle

Evaluation of hierarchical tool-augmented reasoning cannot rely solely on final answer accuracy. It must assess subgoal selection, tool choice, efficiency, safety, and robustness. Process-based evaluation examines whether the agent followed appropriate steps, such as verifying sources or requesting clarification. Outcome-based evaluation examines whether the final result was correct and useful. Both are necessary because a correct answer reached through unsafe means is not acceptable, and a safe process that fails to solve the task is not

sufficient. Human preference data can inform reward models [20], but preference judgments are context-dependent and can encode biases. Evaluation should therefore combine automated metrics, expert review, adversarial testing, and real-world monitoring.

Hierarchical reinforcement learning introduces specific evaluation challenges. Credit assignment across levels can be ambiguous, making it difficult to determine whether a failure originated in the high-level strategy, the mid-level plan, or the low-level execution. Ablation studies can help by removing or replacing individual levels. Counterfactual analysis can examine what would have happened under different subgoals or tools. Stress testing can simulate tool failures, latency spikes, and adversarial inputs. These methods require infrastructure for reproducible experiments and for logging detailed traces. Without such infrastructure, evaluation becomes anecdotal and improvements are difficult to validate.

The operational lifecycle of a tool-augmented hierarchical agent includes design, training, deployment, monitoring, maintenance, and retirement. Each phase has governance requirements. During design, teams should conduct impact assessments and define acceptable use. During training, they should curate data, design rewards, and test for safety. During deployment, they should use staged rollouts and human oversight. During monitoring, they should track performance, fairness, and incidents. During maintenance, they should update models, tools, and policies as environments change. During retirement, they should archive logs, revoke credentials, and migrate users. These activities are iterative, not sequential. Feedback from monitoring should inform retraining and redesign.

Continuous learning is desirable but risky. An agent that updates its policy in production may adapt to new tools and user needs, but it may also drift into undesirable behavior. Reinforcement learning from human feedback can align models with human preferences [3][20], but it can also be gamed. Safe continuous learning requires bounded exploration, conservative update rules, and human review of significant changes. It also requires rollback capability. If a new policy performs worse on safety or fairness metrics, the system should revert to a previous version. The same principle applies to tools: if a tool provider changes its API or terms of service, the system should re-evaluate whether continued use is acceptable.

9. Conclusion

Hierarchical reinforcement learning offers a powerful organizing framework for tool-augmented reasoning in large language models. By separating strategic intent, tactical tool selection, and low-level execution, it can improve long-horizon planning, credit assignment, error recovery, and modular governance. These benefits align with foundational work on temporal abstraction, feudal hierarchies, and data-efficient hierarchical control. They also complement advances in language models, instruction tuning, and tool learning. However, hierarchy is not a panacea. It introduces coordination overhead, new failure modes, and accountability challenges. The value of hierarchical reinforcement learning depends on how it is embedded in a broader system of infrastructure, permissions, monitoring, and policy.

The paper has argued that tool-augmented reasoning should be treated as a socio-technical infrastructure. Robustness requires verification subgoals, fallback mechanisms, and external guardrails. Safety requires permissioned tool access, human oversight for high-risk actions, and scalable oversight mechanisms. Fairness requires attention to distributive and procedural justice, as well as to bias in data and tools. Sustainability requires cost-aware policies and incentives for efficient computation. Governance requires auditability, contestability, and

clear lines of responsibility. Evaluation must be process-oriented as well as outcome-oriented, and it must continue throughout the operational lifecycle.

Future research should develop hybrid hierarchies that combine the semantic flexibility of language models with the stability of reinforcement learning controllers. It should also investigate methods for safe exploration, interpretable subgoal representations, and cross-domain transfer. Policy work should clarify liability, transparency, and data protection in autonomous tool use. Ultimately, the success of hierarchical reinforcement learning for tool-augmented reasoning will be measured not only by benchmark performance but by whether it can be deployed responsibly, equitably, and sustainably in real-world institutions.

References

1. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 5998–6008.
2. 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.
3. 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.
4. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*.
5. Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Zettlemoyer, L., Cancedda, N., & Scialom, T. (2023). Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36.
6. Qin, Y., Hu, S., Lin, Y., Chen, W., Ding, N., Cui, G., Zeng, Z., Zhou, X., Huang, Y., Xiao, C., Han, C., Fung, Y. R., Su, Y., Wang, H., Qian, C., Huang, R., Wu, Z., Liu, P., Liu, Z., & Sun, M. (2023). Tool learning with foundation models. *arXiv preprint arXiv:2304.08354*.
7. Sutton, R. S., Precup, D., & Singh, S. (1999). Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1–2), 181–211.
8. Vezhnevets, A. S., Osindero, S., Schaul, T., Heess, N., Jaderberg, M., Silver, D., & Kavukcuoglu, K. (2017). FeUdal networks for hierarchical reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning* (pp. 3540–3549). PMLR.
9. Nachum, O., Gu, S. S., Lee, H., & Levine, S. (2018). Data-efficient hierarchical reinforcement learning. *Advances in Neural Information Processing Systems*, 31.

10. Kaelbling, L. P., Littman, M. L., & Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1–2), 99–134.
11. Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., & Mané, D. (2016). Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*.
12. Russell, S. (2019). *Human compatible: Artificial intelligence and the problem of control*. Viking.
13. Bender, E. M., Gebru, T., McMillan-Major, A., & Shmitchell, S. (2021). On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency* (pp. 610–623). ACM.
14. Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., Brynjolfsson, E., Buch, S., Card, D., Castellon, R., Chatterji, N., Chen, A., Creel, K., Davis, J. Q., Demszky, D., ... Liang, P. (2021). On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.
15. Weidinger, L., Mellor, J., Rauh, M., Griffin, C., Uesato, J., Huang, P.-S., Cheng, M., Glaese, M., Balle, B., Kasirzadeh, A., Kenton, Z., Brown, S., Hawkins, W., Stepleton, T., Biles, C., Birhane, A., Haas, J., Rimell, L., Hendricks, L. A., ... Gabriel, I. (2021). Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359*.
16. 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.
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. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540), 529–533.
19. Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., Chen, Y., Lillicrap, T., Hui, F., Sifre, L., van den Driessche, G., Graepel, T., & Hassabis, D. (2017). Mastering the game of Go without human knowledge. *Nature*, 550(7676), 354–359.
20. 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.