Reinforcement Learning with World Model Feedback

Andrew Rabinovich

May 5, 2026

Abstract

We stand at the edge of a new challenge in AI: deploying language models, steered by reinforcement learning methods, as agents on tasks that are genuinely complex, multi-dimensional, and subjective. While, standard reinforcement learning methods assume that a single scalar reward is sufficient to guide learning, we argue that this assumption is structurally false for the class of tasks that matters most – those where success requires balancing multiple competing dimensions of quality and where no single ground truth exists against which to verify an output. Scalar feedback cannot tell a model which dimension of its attempt failed, let alone how much that dimension matters. This manuscript advances a position: that effective training on complex tasks requires making the task's dimensional structure explicit within the training process itself, and sustaining that structure across episodes, not just within them. Sub-task decomposition and experiential learning are not independent improvements: the decomposition creates a stable sub-task vocabulary that makes experience indexable and transferable; experiential learning populates that vocabulary with knowledge that compounds over time. Neither is fully valuable without the other – decomposition without accumulated experience means every episode rediscovers the same sub-task strategies from scratch, while experience accumulated without a decomposition collapses into a flat memory that cannot transfer across tasks. We introduce Reinforcement Learning with World Model Feedback (RLWM), a training paradigm built around five pillars: hierarchical decomposition of tasks into weighted sub-components, recurrent structured reflection over those components, relational constraints governing how sub-components interact, stabilization mechanisms that preserve the signal from rare successes, and a persistent graph-structured world model that accumulates experience indexed by the same sub-task vocabulary that structures feedback – enabling what each episode teaches about a competency to transfer across episodes, jobs, and agents. Together, these pillars offer a foundation for agents to learn from experience on the generic tasks that have verifiable rewards.

1 The Limits of Scalar Feedback

Reinforcement learning for language models has made impressive progress on tasks with verifiable answers. When a model attempts a math problem or writes a piece of code, an automated checker can return a binary signal – right or wrong – and policy gradient methods can turn that signal into improved behavior over many episodes. The recent work of Shi et al. [1] on Experiential Reinforcement Learning (ERL) extended this framework by embedding an explicit reflection step into the training loop: after a failed attempt, the model generates a self-critique, then produces a second attempt conditioned on that reflection. Successful second attempts are internalized into the base policy through distillation, so the model eventually learns to perform well without reflection at deployment time. The empirical gains over standard RLVR were substantial, confirming that structured self-reflection is a meaningful lever for learning efficiency.

However, ERL, and scalar-reward RL more broadly, share an assumption that weakens as tasks grow in complexity: the assumption that a single number captures what matters. Consider an agent asked to draft a sustainability plan. Success requires economic feasibility, scientific rigor, social equity, and clarity of communication, all to be achieved simultaneously and in roughly the right proportions. Yet, a single reward for the overall output does not conveys this structure. When the agent's draft fails, the reward cannot say whether the economic projections were unrealistic, whether the scientific evidence was misrepresented, or whether the equity analysis was absent entirely. The model must infer all of this from the direction of a number. This is not a limitation of current implementations; it is an informational constraint. A scalar contains one bit of directional information per attempt. A task with four independent competency dimensions requires at minimum four bits. An exponential amount of exploration can only approximate the

information that the reward signal never contained. This is generally referred to as the credit assignment problem.

The same point applies to ERL's holistic reflection. When the model generates a single self-critique after a failed attempt, it must guess which aspect of the attempt caused the failure. On simple tasks, this guess is often adequate, with only a few ways to go wrong, the reflection converges quickly on the relevant issue. On complex tasks, the guess is frequently wrong or incomplete. The model may fixate on a stylistic flaw while the deeper logical error goes unaddressed. It may over-correct a minor dimension while a critical one remains broken. And because ERL is constrained to a single reflect-retry step per episode, there is no opportunity to course-correct when the first reflection misdiagnoses the problem.

We argue that these are not engineering details to be optimized away; but are fundamental architectural limitations. The path forward is not a better scalar reward or a longer reflection prompt. It is a training framework that makes the task's dimensional structure explicit from the outset, where for each attempt evaluates each sub-task, understanding the competency of dimensions evaluated, which fell short, and how much each one matters.

2 Decomposition and Weighted Reflection

The central focus of RLWM is to treat task structure as a explicit component of the training process. Rather than presenting the model with a task and a single outcome signal, RLWM first decomposes the task into a set of constituent sub-tasks,distinct competency dimensions, each with its own evaluation and its own weight reflecting its relative importance. For a multi-hop question answering task like HotpotQA [2], the sub-tasks might be query formulation, evidence evaluation, and answer synthesis. For a game like Sokoban, they might be spatial reasoning, goal planning, and deadlock avoidance.

The model can propose its own decomposition as part of its chain of thought before the first attempt, or a separate decomposition model can be trained jointly with the main policy to maximize downstream learning efficiency. In established domains, such as medicine or policy analysis, human experts may provide the decomposition the structure of task. The source of the decomposition is a design choice; the commitment to *having* one is not.

Furthermore, there is a deeper reason to care about this commitment beyond any individual task: the vocabulary of sub-tasks is finite, but the tasks that can be constructed from it are not. Consider the range of knowledge-work tasks that humans perform: drafting contracts, writing research proposals, producing clinical summaries, building financial models, developing software specifications. Each is a distinct "job" in the sense that it has its own surface form, its own domain, and its own expected output. But decomposed into competency dimensions, the same sub-tasks recur across all of them: evidence evaluation, logical consistency, audience-appropriate communication, factual accuracy, structural coherence, etc. The jobs are combinatorial compositions of a manageable underlying vocabulary. This means that the number of distinct jobs an agent might face grows exponentially with the number of sub-tasks in the vocabulary, while the number of distinct sub-tasks the agent must actually master remains tractable. An agent trained only at the job level by being rewarded for holistic performance on each task independently, can hardly generalize to new jobs and must effectively relearn from scratch on every new combination. An agent trained at the sub-task level generalizes: mastery of evidence evaluation transfers from legal analysis to scientific summarization to investment research, because the competency is the same even when the surface job is different. This compositionality is not a side effect of decomposition; it is the central argument for why decomposition is necessary in the first place. Without it, the agent's training distribution and its deployment distribution diverge every time a novel job recombines familiar sub-tasks in a new way.

With the decomposition is in place, the reflection phase changes fundamentally. Rather than a single holistic self-critique, RLWM generates a separate reflection for each sub-task, conditioned on that sub-task's specific feedback and on the model's accumulated memory of past successful reflections for similar sub-task dimensions. These per-sub-task reflections are then composed into a prioritized critique ordered by importance weight. Then, the model's attention is directed first toward the dimensions that matter most, and only later toward the minor ones.

Critically, RLWM does not stop after one round of reflection. The reflect-retry cycle is recurrent: the model attempts, receives structured feedback, reflects on each dimension, attempts again, and repeats until

either the attempt meets a success threshold, the budget of learning is exhausted, or improvement stagnates. This recurrent structure matters because complex tasks often cannot be corrected in a single step. A model might address its scientific rigor failure in the first retry, only to discover that the correction introduced a new tension with the economic feasibility dimension. A second reflection can diagnose this new failure; a third can refine it further. The depth of iteration that is appropriate varies by task; RLWM's termination criteria adapt to this automatically, stopping early when gains are no longer being made and continuing when they are.

The best attempt produced during the recurrent loop is then internalized into the base policy through distillation, augmented by a consistency constraint that prevents the update from degrading competencies the model learned on previous tasks. This is the transferability problem that standard RL ignores: when a model improves at policy drafting, it should not inadvertently unlearn the clear reasoning it developed from earlier training on scientific summarization. The consistency constraint ties policy updates across tasks, ensuring that sub-skills once acquired remain available. As such RLWM functions as a multi-task learning system rather than a series of isolated task-specific policies.

3 The Relational Structure of Sub-Tasks

Decomposing a task into sub-tasks raises an immediate question: what governs the relationships between them? Sub-tasks are not independent channels. They interact, conflict, and depend on each other in ways that a flat list cannot represent. The economic feasibility dimension of a policy document and the scientific rigor dimension draw on similar types of evidence; corrective strategies for one should inform the other. The query formulation step of a multi-hop reasoning task feeds directly into the evidence evaluation step; an improvement to the first should not break the second. Some pairs of sub-tasks are fundamentally coupled, improving and degrading together under the same policy changes; treating them independently leads to contradictory updates that cancel impair learning.

We propose that Gestalt principles of perceptual organization [3, 4] provide a principled grammar for specifying the inter-sub-task relationships. The insight from Gestalt psychology – that the whole is different from the sum of its parts, and that the relationships between elements carry information the elements alone do not – applies clearly to multi-dimensional task learning.

- **Proximity** merges elements that are spatially close. In the RLWM setting, sub-tasks that operate on semantically adjacent aspects of a task should be reflected upon jointly, so that their corrective strategies do not conflict. Economic feasibility and scientific rigor are proximate in this sense – both concern factual grounding – and a reflection that strengthens the former by inflating projections while the latter demands conservative estimates cannot serve both. The proximity constraint coordinates their reflections.

- **Good continuation** captures a natural directional flow by grouping elements as part of the same structure. In RLWM, sub-tasks that form a pipeline, where the output of one becomes the input of the next, must respect this directional dependency. Improving query formulation should propagate smoothly downstream to evidence evaluation and answer synthesis; the good continuation constraint penalizes cases where an upstream improvement causes a downstream regression.

- **Closure** captures the requirement that the decomposition cover the entire task. If the global outcome reward diverges substantially from the weighted sum of sub-task rewards, something about the task is not represented in the decomposition, an important dimension has been omitted. This gap is a training signal in its own right, one that can trigger automatic refinement of the decomposition to add the missing coverage.

- **Common fate** addresses coupling between sub-tasks. When two dimensions consistently improve or degrade together under policy changes, they cannot be optimized independently without producing contradictions. The common fate constraint detects these correlations and switches to joint reflection for coupled pairs, while flagging anti-correlated pairs as genuine trade-offs that require explicit balancing via their relative weights.

- **Figure-ground** principle underlines the fact that at any given moment, one sub-task is the primary bottleneck limiting overall performance while others are secondary. Rather than allocating corrective attention uniformly across all dimensions in every iteration, RLWM makes the importance weights state-dependent: whichever sub-task is currently performing worst receives the highest weight in the reflection phase. As that dimension improves, another naturally becomes the new figure, creating an adaptive curriculum within each episode. This is not a heuristic; it is a direct formalization of how effective practitioners actually debug multi-faceted problems.

These constraints are not sub-tasks and they are not rewards. They are an organizing grammar that operates at a level above the decomposition itself, governing how the components of a complex task should interact during learning. Leaving sub-tasks unrelated is analogous to solving a jigsaw puzzle by optimizing each piece independently: the pieces may improve individually while the picture they form grows more incoherent.

4 Accumulated Experience

From Kolb's experiential learning cycle [5] to the replay mechanisms of deep RL [6], there is a persistent theme in learning theory that experience is most valuable when it is retained and reused, not discarded after each episode. ERL made a step in this direction by maintaining a flat memory of successful reflections across episodes. However, a flat memory cannot represent structure: it cannot record that a given corrective strategy worked for scientific rigor but not for social equity, that a particular tool was effective for economic projections but unreliable for evidence synthesis, or that tasks of type A benefit from addressing dimension X before dimension Y.

RLWM organizes accumulated experience as a typed knowledge graph whose nodes represent tasks, sub-tasks, tools, rubrics, and deliverables, and whose edges encode the relationships between them. We call this the Experience Graph. This is the world model that LeCun refers to [9]. There, LeCun argues that effective learning requires a model that operates not in the space of raw observations but in an abstract representation space where the regularities that matter are actually legible. A model that represents the world at the level of tokens faces a state space too vast to generalize from. But, if operated at the right level of abstraction, one can extract transferable structure from sparse experience. For RLWM, that level of abstraction is the sub-task vocabulary. The world the agent needs a model of is not the space of possible jobs but the space of sub-tasks: which strategies reliably produce scientific rigor across task contexts, which tool combinations satisfy economic feasibility, in what order addressing dimensions leads to fastest convergence. The Experience Graph builds exactly this model, not a predictor of future observations, but a structured representation of what the agent has learned about navigating the competency dimensions of complex tasks.

Each node and edge in this graph accumulates its own memory that is not a raw log of past trajectories but structured knowledge of four distinct types.

The first type is task-specific memory: what corrective strategies have worked for a given sub-task dimension in the past, stored and indexed by the sub-task, so that retrieval during reflection is targeted rather than noisy. When the model must improve its scientific rigor in a new episode, it queries the memory attached to the scientific rigor node in the graph and retrieves the most reliable, recent, and frequently successful strategies – not a bag of everything the model has ever reflected on.

The second, is importance-sequencing memory: knowledge about how sub-tasks should be weighted and in what order they should be addressed for a given task structure. Over many episodes, the agent learns not just what the sub-tasks are but in what order tackling them leads to fastest convergence. A sustainability planning task might consistently benefit from addressing scientific rigor first and public narrative last; this curriculum prior is stored on the edges connecting the task to its sub-tasks, and retrieved at the start of future episodes of the same type.

The third type is tool-solving memory: which external tools, and in what combinations, have proven effective for which sub-task types. When an agent operating in an agentic environment discovers that running a Monte Carlo simulation via a code interpreter reliably satisfies an economic feasibility sub-task, this pattern is stored on the edge connecting that sub-task to that tool. Because the index is by sub-task identity rather than task identity, the pattern transfers: when a future task of a different type includes an economic analysis component, the same tool-solving strategy is available.

Lastly, is the rubric-deliverable memory: a growing library of approved and rejected examples for each evaluation dimension. When a human expert approves or rejects a component of the agent's output against a specific criterion, that judgment, along with the salient features of the output in question, is stored on the edge connecting the criterion to the output type. During the next episode's reflection phase, the model retrieves both positive exemplars (concrete examples of what *good* looks like for this criterion) and negative exemplars (examples of what to avoid). The expert's binary feedback is thereby converted into a rich, instance-grounded learning signal that persists across episodes and accumulates into an increasingly refined understanding of the evaluator's standards.

The Experience Graph supports formal operations for consolidation, forgetting, and transfer. Over time, verbose episodic memories are compressed into compact semantic summaries that capture the general principle without the instance-specific noise. This mirrors the transition from episodic to semantic memory in human cognition: early training produces rich, context-specific memories; mature training distills them into reusable heuristics. Memories that are old, rarely retrieved, and low in confidence are pruned, preventing unbounded growth while preserving what has proven useful. When a sub-task has accumulated little experience of its own but resembles a sub-task with rich memory, high-confidence entries can transfer across the similarity edge, allowing the agent to start a novel sub-task with a prior rather than from scratch.

In the limit, the Experience Graph encodes not just what the agent has done, but what the agent has learned: which strategies work for which competency dimensions, with which tools, under which evaluation criteria, and in what order. This is the difference between an agent that merely records experience and an agent that genuinely learns from their own and other's experiences.

The most important property of the Experience Graph is one that emerges when multiple agents are operating simultaneously: the sub-task vocabulary makes the graph shareable. Since memory is indexed by sub-task identity rather than by job identity, an Experience Graph built from many agents working on many different jobs remains internally coherent. Agent A, working on a legal contract review job, may fail on its evidence evaluation sub-task and write a corrective reflection to the evidence evaluation node. Agent B, simultaneously working on a medical literature synthesis job, has the same evidence evaluation node in its decomposition. Without any explicit coordination, Agent B's next reflection on that sub-task can be informed by Agent A's failure – not because the jobs are similar, but because the competency is shared.

This cross-agent transfer is qualitatively different from what any single agent can achieve on its own. A lone agent encounters evidence evaluation in some jobs and not others, succeeds sometimes and fails sometimes, and accumulates a sample of experience that is both sparse and biased by whatever job distribution it happened to train on. A population of agents, each writing their failures and successes to the same shared sub-task nodes, collectively generates a dense, diverse sample of experience for every sub-task in the vocabulary – orders of magnitude richer than any individual agent could produce. The Experience Graph therefore functions as a collective intelligence: a shared medium through which agents propagate lessons to each other across job boundaries that, from the outside, appear entirely unrelated.

This also strengthens the consistency constraint beyond its within-agent role. It is not enough that a single agent's policy on evidence evaluation remains stable as it trains on new jobs. For cross-agent transfer to be meaningful, the sub-task rewards themselves must be consistent across agents. A successful evidence evaluation in a legal context must be commensurable with a successful evidence evaluation in a medical context, or the shared node accumulates conflicting signals that degrade rather than improve retrieval. This cross-agent consistency requirement is a constraint on the rubric design, it is how human evaluators are trained and calibrated across different job domains and is a constraint on the learning algorithm. Getting it right is a precondition for the Experience Graph to function as collective memory rather than as a divergence of incommensurable feedback.

5 The Signal Problem: Sub-task Decomposed Advantage Estimation

Standard policy gradient methods assign the same advantage to every token in a response: whatever scalar the group-relative normalization produces for the trajectory as a whole, every token in that trajectory is trained on it uniformly. For single-objective tasks this is defensible. For multi-sub-task responses it is not, as different sub-tasks have their decisive moments at different positions. The retrieval quality of a response

may be determined in the early turns; the output formatting may be determined in the late turns. A scalar broadcast to all tokens treats every position as equally responsible for outcomes that most of them barely influenced. When sub-tasks disagree in sign, the scalar averages to near zero and the optimizer trains on nothing at all, despite the response containing a rich, spatially differentiated signal.

RLWM addresses this through Sub-task Decomposed Advantage Estimation (SDAE). The key observation is that each sub-task has a pivot: the position in the response where the generation choices most determined whether that sub-task would succeed or fail. SDAE assigns each token a per-token advantage that is the importance-weighted sum of sub-task signals, each localized around its sub-task's pivot and decaying smoothly away from it. The credit signal thereby inherits the same dimensional priorities as the reflection phase, and gradient mass concentrates near the decisive region for each competency rather than diffusing uniformly across the response.

The improvement over scalar broadcasting is most visible when sub-tasks disagree. When their advantages have opposite signs and their pivots are far apart, the broadcast scalar collapses to near zero. SDAE on the other hand, reinforces the tokens responsible for the success and penalizes the tokens responsible for the failure, thus carrying information that no scalar can express. When sub-tasks agree in sign but differ in magnitude, cancellation is avoided but scalar broadcasting still allocates credit uniformly, but SDAE concentrates it toward the dominant sub-task's pivot. When one sub-task is strongly positive and another mildly negative, the broadcast scalar implies uniform reinforcement and conceals a weak penalty; SDAE preserves both lobes. In all three cases, the per-token shape carries information about which part of the response determined each dimension's outcome – information a scalar discards by construction. In a rare case where sub-tasks agree completely, SDAE degrades gracefully to the scalar case, imposing no cost in regimes where per-token shaping is not needed.

6 Beyond Verifiable Answers

The deeper motivation for RLWM is not to improve performance on the benchmarks where scalar-reward RL already works reasonably well. It is to extend reinforcement learning to the class of tasks that has so far resisted it: tasks where there is no algorithmic verification, where success is defined by structured human judgment, and where the evaluation dimensions are multiple, interdependent, and of differing importance.

This is not a niche class of tasks. It describes the majority of economically and socially significant work that humans do: writing, analysis, design, planning, teaching, clinical judgment, scientific reasoning, policy development. Current AI systems are increasingly deployed on tasks of this type, and they are increasingly evaluated against scalar proxies for what actually matters. These proxies are chosen not because they capture task quality but because they can be computed automatically. The result is a systematic misalignment between what the training signal optimizes and what the deployment context requires. In these domains, AI models that excel at academic benchmarks, therefore, perform poorly in real world scenarios.

RLWM takes a different path. Rather than approximating multi-dimensional evaluation with a scalar proxy, it embraces the multi-dimensional structure directly. The key insight is that evaluation is *strictly easier than generation* – and this asymmetry is not incidental but fundamental.

In complexity theory, the distinction between solving a problem and verifying a proposed solution is one of the deepest in all of mathematics. NP problems may be computationally intractable to solve, yet any candidate solution can be checked efficiently. A human expert asked to draft a rigorous sustainability plan from scratch faces exactly this kind of difficulty: the generation problem is hard, time-consuming, and demands rare expertise. But the same expert, presented with a specific claim in a specific sub-section of an agent-generated draft and asked *does this economic projection meet the standard of rigor?*, faces a verification problem. Recognition precedes and is easier than generation. An expert can reliably identify that an argument is logically valid without being able to produce it; can judge that an evidence summary is balanced without being able to write it; can determine if a video game is interesting and bug free without being able to create one. This is not a limitation of the expert, it is the structure of the problem.

RLWM is designed to exploit this asymmetry systematically. The training signal does not ask human experts to generate better outputs or to explain how to improve the agent's attempt. It asks only for a binary judgment on a focused competency dimension: does this sub-task response meet the criterion or not? This is the minimal verification task – and it is, by the logic above, tractable even when the corresponding

generation task is not. The per-sub-task decomposition is what makes verification tractable at scale: an evaluator asked to judge an entire policy document faces a hard, holistic problem, while evaluator asked to judge only the scientific rigor of one section faces a focused, well-scoped one. Sub-task decomposition does not merely improve the training signal by adding structure – it transforms an intractable evaluation problem into a tractable one by reducing it to a sequence of targeted verification steps.

Over many episodes, this structured signal accumulates in the Experience Graph as rubric-deliverable memories that progressively sharpen the model's understanding of each evaluator's standards. The agent, in effect, learns to solve problems in ways that can be verified but not easily generated by observing which of its many attempts pass the verification threshold. It does so far more efficiently than any approach that demands human generated training signal.

Silver and Sutton [8] have argued that the next scaling regime for AI will come from agent-generated data through interaction rather than from static human text – that experience, not imitation, is the medium through which truly novel capabilities will emerge. RLWM is an architectural proposal for what the training machinery of that era needs to look like when the tasks are complex and the rewards are structured. It does not replace the need for grounded rewards and rich action spaces; it provides the reflection scaffolding, the relational constraints, and the accumulated memory that make experiential learning efficient once those ingredients are in place.

The challenge that remains is at the frontier where neither verifiable environments nor human evaluators can provide reliable feedback. An agent must act in a non-ergodic real-world system and observe consequences that no one has pre-specified. Reaching that frontier will likely require integrating RLWM with embodied and agentic settings in which the agent can design experiments, observe outcomes, and discover knowledge that no existing evaluation rubric could have anticipated. That frontier is not reachable until the nearer one is crossed: learning reliably and efficiently on complex tasks where structured human judgment is available. That is the problem RLWM addresses.

7 Conclusion

The central claim of this paper is straight forward: scalar feedback is structurally insufficient for training agents on complex, multi-dimensional tasks. The architectural response to this insufficiency must be built into the training framework itself. Leaving the structure implicit by hoping the model infers such a *world model* from a single number is not a recoverable limitation in general. This is a design choice that accepts a systematic ceiling on what can be learned.

RLWM is a proposal for what that design should look like instead. Tasks are decomposed into weighted sub-components so that feedback is dimensionally faithful. Reflection is structured and recurrent so that the model can diagnose specifically what failed and iterate until it is corrected. Gestalt relational constraints govern how sub-tasks interact, preventing the incoherence that arises from treating independent dimensions as truly independent. Sub-task Decomposed Advantage Estimation localizes the credit signal to each sub-task's decisive region, ensuring that structured feedback translates into a training signal whose per-token shape matches the spatial reality of where each competency succeeded or failed. And the Experience Graph accumulates what is learned – strategies, orderings, tool patterns, evaluator standards – into a persistent knowledge structure that improves with every episode rather than starting from scratch.

Two properties of this framework deserve particular emphasis. First, the sub-task vocabulary is finite even though the space of jobs is not. Mastering a manageable set of competency dimensions is sufficient to generalize across an exponentially large space of real-world tasks, because most jobs are permutations of known sub-tasks in new proportions and new contexts. This compositionality is what transforms RLWM from a method for training on specific tasks into a method for training agents that generalize. Second, the sub-task vocabulary is shared even when the agents are not. A population of agents working on different jobs can pool their experience at the sub-task level and one agent's failure on evidence evaluation informs every other agent encountering that same competency regardless of what job it appeared in. This collective memory is only possible because the sub-task index creates a basis language across otherwise incomparable jobs. Getting the consistency of that index right by ensuring that sub-task rewards mean the same thing across agents, evaluators, and domains – is the foundational requirement that makes everything else work.

Acknowledgements. The author thanks the authors of ERL [1] and R^3L [7] for foundational work that RLWM

builds upon.

References

[1] T. Shi, S. Chen, B. Jiang, L. Song, L. Yang, and J. Zhao. Experiential Reinforcement Learning. *arXiv:2602.13949*, 2026.

[2] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. Cohen, R. Salakhutdinov, and C. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*, 2018.

[3] M. Wertheimer. Untersuchungen zur Lehre von der Gestalt, II. *Psychologische Forschung*, 4(1):301–350, 1923.

[4] K. Koffka. *Principles of Gestalt Psychology*. Harcourt, Brace & Company, 1935.

[5] D. A. Kolb. *Experiential Learning: Experience as the Source of Learning and Development*. FT Press, 2nd edition, 2014.

[6] V. Mnih, K. Kavukcuoglu, D. Silver, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[7] W. Shi, Y. Chen, Z. Li, X. Pan, Y. Sun, J. Xu, X. Zhou, and Y. Li. R^3L: Reflect-then-Retry Reinforcement Learning with Language-Guided Exploration, Pivotal Credit, and Positive Amplification. *arXiv:2601.03715*, 2026.

[8] D. Silver and R. S. Sutton. Welcome to the Era of Experience, 2025.

[9] Y. LeCun. A Path Towards Autonomous Machine Intelligence. OpenReview preprint, 2022.