
Bulletproof: LLM Reasoning Enhancement via Classical Reinforcement Learning

Dinesh Vasireddy
Computer Science
Harvard University
dineshvasireddy@college.harvard.edu

Abstract

Recent advances in large language models (LLMs) have yielded impressive generative capabilities, yet robust multi-step reasoning remains a fundamental challenge, especially on benchmarks such as Humanity’s Last Exam (HLE). We present Bulletproof, a reinforcement learning (RL) framework that enhances the reasoning abilities of open-source LLMs by simulating structured reasoning tokens—such as `<think>`, `<verify>`, and `<conclude>`—using Proximal Policy Optimization (PPO). Our approach rewards models for logical consistency, stepwise correctness, and factual accuracy, while penalizing hallucinations and unsupported claims, among other factors, all without requiring large-scale human-annotated datasets. We evaluate Bulletproof on a diverse subset of HLE, demonstrating that RL-based reasoning token simulation yields measurable improvements in logical coherence and answer accuracy over baseline models, with up to 0.8% absolute accuracy gain and a 0.05 increase in composite reward. Our analysis reveals that while structured reasoning enforcement is promising, further optimization of reward functions and hallucination penalties is necessary to achieve substantial gains. These results suggest that classical RL can bridge the gap between pattern-matching and genuine reasoning in LLMs, providing a scalable path toward more reliable and interpretable AI systems.

1 Introduction

The pursuit of robust reasoning in artificial intelligence has long been a central challenge, with implications spanning mathematics, science, law, and beyond. While numerous flagship large language models (LLMs) such as GPT-4o, Claude 3.5-Sonnet, and Llama 2/3 have achieved remarkable fluency and versatility, their ability to perform complex, most of their multi-step reasoning has remained limited since their inception. This shortfall is especially apparent on rigorous benchmarks like Humanity’s Last Exam (HLE), which are designed to probe not just surface-level pattern recognition, but genuine logical inference and structured problem-solving [4, 1].

However, a new class of “reasoning” models—such as OpenAI’s o1 and o3 series, and DeepSeek-R1—has emerged within the past year, aiming to address these limitations by explicitly targeting logical consistency and stepwise problem-solving. While these models have demonstrated some improvements, their performance on HLE and similar benchmarks still lags far behind human-level reasoning, with even the best models rarely exceeding 15% accuracy [2, 4]. This persistent gap highlights the need for new approaches that go beyond surface-level fluency and pattern matching.

Traditional strategies for improving LLM reasoning, such as Chain-of-Thought (CoT) prompting and supervised fine-tuning on annotated reasoning traces, have shown some promise but are fundamentally limited. Prompting methods do not alter the model’s internal reasoning process, and supervised fine-tuning requires large, high-quality datasets that are expensive to construct and often fail to

generalize across domains [3, 7]. As a result, there is growing interest in reinforcement learning (RL) as a scalable alternative for enhancing reasoning in LLMs. RL enables models to learn from structured feedback, optimizing for properties such as logical coherence, factual accuracy, and self-correction through reward signals [2, 6].

Yet, the application of RL to LLM reasoning is far from straightforward. Key questions remain: How should reward functions be designed to encourage not just format compliance, but genuine logical progress? Can RL-based methods reduce hallucinations and unsupported claims, or do they risk introducing new pathologies? And, crucially, can these methods bridge the gap between the pattern-matching tendencies of current models and the robust, stepwise reasoning required for tasks like those in HLE?

In this work, we explicitly address the following research question:

How can simple reinforcement learning be used to simulate robust reasoning tokens in base open-source language models (Phi-2, GPT-2, TinyLlama, etc.) to improve their performance on complex reasoning tasks, such as those in Humanity’s Last Exam (HLE)?

Our central hypothesis is that by rewarding models for logical consistency, stepwise correctness, and factual accuracy—while penalizing hallucinations and unsupported claims—reinforcement learning can move LLMs beyond superficial format compliance and foster genuine advances in reasoning ability. To test this, we introduce Bulletproof, a lightweight reinforcement learning framework that simulates structured reasoning tokens in open-source LLMs using Proximal Policy Optimization (PPO), and systematically evaluate its impact on HLE performance.

By rigorously analyzing the effects of RL-based reasoning token simulation, we aim to shed light on both the potential and the limitations of this approach, and to provide practical insights for the design of more reliable and interpretable AI systems.

Table 1: Example Generated Output for HLE Question with TinyLlama-1.1B

Field	Content
Question	Suppose X is a compact connected metric space. We say X is connected im kleinem at $x \in X$ to mean that every neighborhood of x contains a connected neighborhood of x . Suppose X fails to be connected im kleinem at at least one point. What is the smallest possible cardinality of the set of points where X fails to be connected im kleinem?
$\langle think \rangle$	The question involves understanding the concept of 'connected im kleinem' in the context of a compact connected metric space. The key concepts are: 1) compactness, 2) connectedness, and 3) the specific definition of connected im kleinem. The reasoning steps include analyzing what it means for a space \dots which may imply a certain structure or separation in the space. The question asks for the smallest cardinality of the set of points where this failure occurs, leading to the consideration of whether one point is sufficient or if more points are necessary to maintain the overall connectedness of the space.
$\langle verify \rangle$	To verify the reasoning, we consider the definition of connected im kleinem and the implications of a space being connected. If a space is connected but fails to be connected im kleinem at one point, it suggests that there is a local disconnection at that point. \dots Therefore, we check if having just one point is consistent with the definition and does not contradict the properties of compact connected metric spaces. This leads to the conclusion that one point is indeed the minimum required to illustrate the failure of connected im kleinem.
$\langle conclude \rangle$	1
HLE Answer	0
Reward	0.4083

2 Related Works

The intersection of reinforcement learning (RL) and large language model (LLM) reasoning has rapidly become a focal point in AI research, driven by the need to move beyond surface-level fluency toward robust, stepwise logical inference. Here, we review the most relevant literature across RL-based fine-tuning, reward function design, evaluation benchmarks, hallucination mitigation, and persistent challenges, situating the Bulletproof project within this evolving landscape.

2.1 Reinforcement Learning for LLM Reasoning

The use of RL to align LLMs with human preferences and reasoning objectives was popularized by Ouyang et al. (2022), who introduced Reinforcement Learning from Human Feedback (RLHF) using Proximal Policy Optimization (PPO) to fine-tune GPT-3, resulting in the widely adopted InstructGPT model [8]. This approach established PPO as a standard for post-training alignment, demonstrating that RL can improve truthfulness and reduce toxicity while making models more responsive to instructions. Building on this, Rafailov et al. (2023) proposed Direct Preference Optimization (DPO), a lightweight alternative to PPO that achieves similar or better alignment by reparameterizing the reward objective as a single-step loss [9]. DPO’s stability and efficiency suggest that much of PPO’s benefit for reasoning can be captured with simpler objectives.

Recent work has also shown that RL can be effective even for small, resource-constrained models. Dang and Ngo (2025) demonstrated that a distilled 1.5B parameter model, fine-tuned with a PPO variant (GRPO) and only 7k training examples, could achieve rapid gains on math reasoning benchmarks, surpassing OpenAI’s o1-preview model at a fraction of the cost [10]. Havrilla et al. (2024) provide a comprehensive comparison of RL fine-tuning algorithms for LLM reasoning, showing that PPO and Expert Iteration both improve multi-step reasoning, but that RL training often does not explore beyond solutions found by supervised baselines [11]. Han et al. (2023) show that dialogue-guided chain-of-thought (DialCoT) with PPO can decompose complex problems into sub-questions, enabling smaller models to achieve state-of-the-art results on arithmetic reasoning [12].

A major milestone in RL for LLM reasoning is DeepSeek-R1 [2], which uses large-scale RL (with and without supervised warm-up) to unlock complex reasoning behaviors. DeepSeek-R1-Zero, trained purely via RL from scratch, achieved strong reasoning but required subsequent supervised fine-tuning for fluency. The final DeepSeek-R1 model reaches parity with OpenAI’s o1 on reasoning tasks, and the open-sourcing of distilled models (1.5B–70B) highlights the scalability of RL-based approaches. Symbolic and tool-assisted feedback is another promising direction: Jha et al. (2024) introduced Reinforcement Learning via Symbolic Feedback (RLSF), where LLMs are fine-tuned using verifiable feedback from theorem provers or code testers rather than fuzzy reward models [13].

2.2 Reward Function Design and Hallucination Mitigation

Reward function design is central to the success of RL-based LLM fine-tuning. Liu et al. (2024) introduced logical consistency scoring as a reward mechanism, penalizing contradictions and encouraging coherent reasoning steps [3]. Sarukkai et al. (2025) further advanced this by automating reward generation using task-specific progress functions, reducing reliance on human annotation [6]. Rita et al. (2024) addressed reward over-optimization by calibrating rewards with human demonstrations, mitigating the risk of models exploiting poorly designed objectives [5]. Zhang et al. (2024) introduced Chain-of-Preference Optimization (CPO), using tree search to guide LLM reasoning and align each step with high-quality reasoning paths [14]. Lee et al. (2024) proposed CREST, a consistency-driven rationale evaluation method for self-training LLMs to produce logically consistent reasoning [15].

Several works have focused on hallucination and factuality. Nakano et al. (2022) developed WebGPT, which uses RLHF to train models to issue search queries and cite evidence, with a reward model favoring accurate, well-referenced answers [16]. Bai et al. (2022) introduced Constitutional AI, where a model-generated preference model (no humans in the loop) rewards outputs that adhere to a set of principles, using AI-generated critiques to curb toxic or illogical outputs [17]. Farquhar et al. (2024) proposed an entropy-based uncertainty measure to detect hallucinations in LLMs [18].

Recent advances also address calibration and self-evaluation. Wang et al. (2024) proposed CREAM, a regularization technique for self-rewarding LLMs that maximizes reward consistency across iterations, stabilizing self-training and preventing drift [19]. Stangel et al. (2024) tackled hallucinations from a

calibration perspective, training models via RL to output calibrated confidence scores and rewarding doubt, which helps models avoid confidently incorrect answers [20].

2.3 Evaluation Benchmarks for Reasoning

The evaluation of reasoning in LLMs has evolved with the introduction of more challenging and diagnostic benchmarks. The ARC Prize (ARC-AGI-2) [1] is currently the most prominent and unbeaten benchmark for abstract reasoning and general intelligence in AI, with a global competition and leaderboard that includes all major reasoning models and research labs. ARC-AGI-2 is specifically designed to stress test the efficiency and capability of state-of-the-art AI reasoning systems, requiring compositional, contextual, and symbolic reasoning that remains easy for humans but extremely difficult for AI. Humanity’s Last Exam (HLE) [4] builds on the tradition of ARC, expanding the evaluation to 2,500+ expert-crafted questions across diverse disciplines. Both ARC and HLE expose the limitations of current models, with even the best systems scoring far below human experts. LogicGame [23] and Math-RoB [24] further probe rule-based reasoning and robustness, revealing that many purported reasoning gains are brittle. PlanBench [25] and BIG-Bench Hard [26] focus on planning, multi-step reasoning, and adversarial robustness, while TruthfulQA [27] tests adversarial truthfulness and factual consistency.

2.4 Reasoning-Focused LLM Models and Architectures

Recent years have seen the emergence of LLMs explicitly designed for reasoning. DeepSeek-R1 [2] is an open-source model trained for high-level reasoning using RL, achieving performance on par with proprietary models like OpenAI’s o1. Lewkowycz et al. (2022) introduced Minerva, a specialized LLM for mathematical and scientific reasoning, showing that targeted training on high-quality reasoning data enables rigorous, stepwise solutions [28]. These models set new bars for reasoning-heavy benchmarks and demonstrate the value of structured reasoning data and RL-based fine-tuning.

2.5 Surveys and Synthesis

Comprehensive surveys such as Wu (2025) [29] provide an overview of post-training alignment techniques for LLMs, including RLHF, reward-assisted decoding, and self-correction. These works emphasize that reward design is now a key lever for eliciting reasoning, and highlight challenges such as reward hacking, interpretability, and the need for grounded, externally verifiable rewards.

2.6 Landscape

The existing research landscape demonstrates that RL-based fine-tuning—especially with carefully designed, multi-faceted reward functions—can substantially improve the reasoning capabilities of LLMs. However, most prior work has focused on large, resource-intensive models or required extensive human feedback, limiting accessibility and scalability. In contrast, Bulletproof is designed to bring structured reasoning enhancements to lightweight, open-source models, with the explicit goal of making robust, interpretable reasoning accessible to a broader range of systems. By targeting models that typically perform poorly on challenging benchmarks like HLE, our framework addresses a critical gap: enabling low-cost, scalable reasoning improvements for weaker models that are often overlooked in the literature. Unlike approaches that rely on massive supervised datasets or proprietary architectures, Bulletproof leverages PPO-based RL and automated, multi-component reward functions to induce stepwise reasoning in any base model, without the need for extensive annotation or computational resources. By systematically evaluating on HLE and related benchmarks, Bulletproof not only demonstrates the feasibility of this approach but also provides new insights into the design and evaluation of RL-enhanced reasoning for the next generation of accessible, high-utility LLMs.

3 Methodology

Our research methodology is centered on enhancing the multi-step reasoning capabilities of open-source Large Language Models (LLMs) through a novel reinforcement learning (RL) framework.

This framework utilizes Proximal Policy Optimization (PPO) to train LLMs to effectively generate and utilize structured reasoning tokens. The core of our approach lies in a multi-faceted reward function designed to promote logical consistency, stepwise correctness, factual accuracy, and to penalize hallucinations. We evaluate our methodology on the challenging Humanity’s Last Exam (HLE) benchmark.

Our framework operates by fine-tuning pre-trained LLMs using PPO. The key innovation is the introduction of explicit reasoning tokens that guide the model to structure its thought process into distinct phases: thinking, verification, and conclusion. An auxiliary LLM (specifically, GPT-4.1 in our implementation) is employed to parse the generated text containing these tokens, extracting the content of each reasoning phase. This parsed information is then fed into a comprehensive reward model, which calculates a scalar reward signal used to update the policy of the primary LLM.

3.1 Base Model Selection and Adaptation with Prompting

We select six lightweight open-source LLMs for our experiments to assess the generalizability of our approach across different model architectures and sizes (Table 2).

Table 2: Lightweight language models and their HuggingFace identifiers

Model	HuggingFace Code
tinylama	TinyLlama/TinyLlama-1.1B-Chat-v1.0
phi	microsoft/phi-2
stablilm	stabilityai/stablilm-3b-4e1t
flan_t5_small	google/flan-t5-small
gpt2	openai-community/gpt2
opt	facebook/opt-1.3b

A crucial pre-processing step involves adapting these base models to understand and generate structured reasoning tokens. We introduce special tokens into the vocabulary of each model: `<think>`, `</think>`, `<verify>`, `</verify>`, `<conclude>`, and `</conclude>`. The model embeddings are resized to accommodate these new tokens. During generation, models are prompted to use this structure, thereby externalizing their reasoning process. The content within `<think>...</think>` is intended to capture the model’s preliminary reasoning steps, `<verify>...</verify>` for self-correction and fact-checking, and `<conclude>...</conclude>` for the final answer.

3.2 Proximal Policy Optimization (PPO)

We employ Proximal Policy Optimization (PPO) [21] to fine-tune the base LLMs. PPO is a policy gradient method known for its stability and sample efficiency, making it well-suited for LLM fine-tuning. In our PPO implementation:

- **Policy and Value Functions:** The LLM itself serves as the policy network $\pi_\theta(a_t|s_t)$, which outputs a probability distribution over the next token given the current context (question and previously generated tokens). A separate, learned value function $V_\phi(s_t)$ estimates the expected future reward from state s_t . This value function is typically a small neural network head attached to the LLM.
- **Advantage Estimation:** We use Generalized Advantage Estimation (GAE) [22] to compute the advantage \hat{A}_t , which reduces variance in policy gradient estimates.
- **Objective Function:** The PPO objective function aims to maximize the expected reward while constraining the policy update to stay close to the previous policy, typically using a clipped surrogate objective:

$$L^{CLIP}(\theta) = \mathbb{E}_t[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon)\hat{A}_t)]$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ is the probability ratio and ϵ is a hyperparameter (e.g., 0.2) defining the clipping range. An entropy bonus is often added to encourage exploration.

- **Training Loop:** The training involves iteratively sampling trajectories (i.e., generating responses to HLE questions), calculating rewards using our comprehensive reward function, computing advantages, and updating the policy and value function parameters using an optimizer like AdamW.

This framework ultimately climbs towards structured reasoning in model outputs by employing the following approach. Rather than imposing a hard constraint on the model’s output, our PPO framework incentivizes the use of structured reasoning tokens through a carefully designed reward mechanism. Specifically, the input prompts explicitly instruct the model to organize its response using the `<think>`, `<verify>`, and `<conclude>` tokens, delineating distinct phases of reasoning.

After generation, each response is automatically parsed to detect the presence and correct ordering of these tokens using deterministic pattern-matching algorithms. Outputs that fail to include all required token pairs, or that misuse the structure (e.g., missing a section or incorrect nesting), are assigned minimal or negative rewards across multiple components of the reward function: logical consistency (empty or trivial reasoning graph), stepwise correctness (no verifiable steps), and answer correctness (absence of a structured conclusion).

The policy gradient optimization process naturally converges toward policies that reliably produce well-structured, token-delimited reasoning in accordance with our restrictions, since only such outputs are consistently associated with high rewards. This approach enables the model to internalize the structured reasoning protocol as an emergent property of reward maximization, rather than as a brittle, externally imposed constraint.

Our implementation leverages the Hugging Face Transformers library for model handling and TRL (Transformer Reinforcement Learning) for the PPO training loop, adapting it to incorporate our custom reasoning token generation and reward calculation pipeline.

3.3 Reward Function Design

The efficacy of our RL approach hinges on a meticulously designed reward function R . This function provides a scalar signal that guides the LLM towards generating more structured, coherent, and correct reasoning. The overall reward is a weighted sum of several components:

$$R = w_c C_{\text{logic}} + w_s S_{\text{step}} + w_a A_{\text{ans}} - w_h H_{\text{halluc}}$$

where w_c, w_s, w_a, w_h are tunable hyperparameters representing the weights for each component.

1. **Logical Consistency (C_{logic}):** This component assesses the coherence and logical flow within the `<think>` and `<verify>` sections. Our implementation extracts individual reasoning steps from the `<think>` block. These steps are then structured into a directed graph using the NetworkX library. We evaluate properties such as acyclicity (penalizing circular reasoning) and connectivity. A simple heuristic rewards well-formed, non-trivial reasoning graphs. For instance, a sequence of causally linked, non-contradictory statements receives a positive score, while disconnected or contradictory statements are penalized.
2. **Stepwise Correctness (S_{step}):** This measures the validity of intermediate reasoning steps and the effectiveness of the self-correction process within the `<verify>` block. The content of the `<verify>` block is compared against the steps outlined in the `<think>` block. If the verification process identifies and corrects a flaw or confirms a step using evidence (simulated or actual), it receives a positive reward. Our implementation uses semantic similarity (via sentence embeddings) and rule-based checks (e.g., for mathematical operations if applicable to the question type) to assess the quality of verification.

Table 3: Illustration of Stepwise Correctness Evaluation: Mapping Reasoning Steps to Verification

Reasoning Step (<think>)	Verification (<verify>) Reference	Verified?
1. The equation $2x + 3 = 7$ can be rearranged to $2x = 4$.	The step where $2x + 3 = 7$ is rearranged to $2x = 4$ is correct.	✓
2. Dividing both sides by 2 gives $x = 2$.	The division by 2 to get $x = 2$ is valid.	✓
3. The solution $x = 2$ is the final answer.	(No explicit reference in verification.)	×

3. **Answer Correctness (A_{ans}):** This evaluates the accuracy of the final answer provided in the <conclude> block against the ground truth from the HLE dataset. Our implementation uses a large language model (LLM) to assess the candidate answer, returning a nuanced confidence score based on correctness, partial correctness, and detailed justification. This allows for graded scoring rather than a simple binary match. If the LLM-based evaluation is unavailable, we fall back to exact match for multiple-choice questions and embedding similarity for open-ended answers.
4. **Hallucination Penalty (H_{halluc}):** This component penalizes the model for generating unsupported claims or factually incorrect statements within its reasoning process. Our implementation employs a multi-pronged approach:
 - **Pattern-based detection:** We use regular expressions to identify common hallucination patterns (e.g., overly confident assertions, false citations, definitive statements about unverifiable information).
 - **Relevance to question:** The semantic similarity between the reasoning text (content of <think> and <verify>) and the original question is calculated. Low relevance, indicating the model has diverged significantly, incurs a penalty.
 - **(Stretch Goal/Future Work) Retrieval-based verification:** For certain domains, reasoning steps could be cross-checked against an indexed knowledge base (e.g., a FAISS index of relevant factual snippets). Mismatched or unsupported claims would be penalized. This was outlined in the checkpoint but is a more complex component.

The aggregated hallucination score is then subtracted from the total reward.

The weights w_i are critical and are tuned empirically through experimentation on a validation set to balance the different aspects of good reasoning.

3.4 Dataset and Experimental Setup

We utilize the Humanity’s Last Exam (HLE) dataset [4] for both fine-tuning and evaluation. HLE comprises *2500 expert-level questions across physics, mathematics, chemistry, history, philosophy, etc.* There are primarily two types of questions: multiple choice and short response. These questions often require complex multi-step reasoning, detailed in the preceding literature review [4], making it an ideal testbed for our methodology.

- **Data Splitting:** The HLE dataset is split into training (70%), validation (15%), and test sets (15%). The training set is used for PPO fine-tuning. The validation set is used to monitor model performance during training, enabling the detection of overfitting and providing an unbiased estimate of generalization throughout the optimization process. The test set is held out for final performance evaluation.
- **Baselines:** For each model, we first evaluate the base pre-trained model (prior to any fine-tuning) on the test set using our structured reasoning prompt and reward-based metrics. After fine-tuning with PPO, we re-evaluate the same model on the test set. The primary comparison is between the baseline (pre-trained) and fine-tuned model performance, quantifying the improvements attributable to our reinforcement learning approach.
- **Training Infrastructure:** Training is conducted using PyTorch and the Hugging Face ecosystem on available GPU resources.

3.5 Evaluation and Measuring Success

To rigorously assess the impact of our reinforcement learning framework, we adopt a holistic evaluation strategy that goes beyond simple accuracy or exact match scores. Our primary approach is to measure the change in overall reward—computed via our multi-factor reward function—before and after PPO fine-tuning, across all models and question types. This reward aggregates logical consistency, stepwise correctness, answer accuracy, and hallucination penalties, providing a nuanced view of model reasoning quality.

We report not only the aggregate reward improvements but also analyze the evolution of individual subcomponents, such as logical consistency and stepwise correctness, to understand which aspects of reasoning are most affected by fine-tuning. Trends in these metrics are examined both within and

across models, revealing whether improvements are broad-based or isolated to specific capabilities. For instance, we observe that models in the 1–2B parameter range tend to show the most pronounced and balanced gains, while smaller models may struggle with consistency and larger models may approach a performance ceiling.

Success in our framework is defined by consistent, statistically significant improvements in overall reward and its subcomponents across multiple models, rather than isolated gains in a single metric or model. We consider the approach effective if all or most models demonstrate positive shifts in aggregate reward and key submetrics after fine-tuning, if improvements extend beyond answer correctness to include reasoning structure and reduced hallucinations, and if the relative balance between metrics is preserved or improved, indicating that fine-tuning enhances performance holistically rather than overfitting to a single aspect of the reward.

In addition to quantitative results, we conduct qualitative analysis of generated reasoning traces, examining the structure, relevance, and interpretability of model outputs before and after fine-tuning. This dual focus on quantitative trends and qualitative reasoning quality allows us to draw robust conclusions about the effectiveness and generalizability of our method. Ultimately, a successful outcome is one in which reinforcement learning with structured reasoning tokens yields broad, interpretable, and reliable improvements in LLM reasoning performance across diverse models and tasks.

4 Results

We include all code, results, scripts, etc. in our github repo linked in Appendix Figure 5.

4.1 Consistent Reward Gains and the “Sweet Spot” Effect

PPO-based fine-tuning with structured reasoning tokens led to consistent and statistically significant improvements in overall reward across all tested models. For example, TinyLlama-1.1B improved its overall reward from 0.3879 to 0.4676 (+0.0797, +20.5%), while Flan-T5-Small increased from 0.3238 to 0.3710 (+0.0473, +14.6%). The smallest improvement was observed in GPT-2, which increased from 0.3121 to 0.3422 (+0.0301, +9.6%). On average, models improved their overall reward by approximately 12–20%. As shown in Figure 1, every model, regardless of architecture or size, exhibited a positive shift in aggregate reward after fine-tuning. Notably, the magnitude of improvement was not uniform: models in the 1–2B parameter range (TinyLlama-1.1B, OPT-1.3B, Phi-2) demonstrated the largest and most balanced gains. This trend is captured quantitatively by the quadratic fit in Figure 2, which achieves a high coefficient of determination ($R^2 = 0.87$), indicating that model size explains a substantial portion of the variance in reward improvement. Statistically, this supports the existence of a “sweet spot” for model capacity, where models of moderate size are best positioned to benefit from structured reasoning fine-tuning, while smaller models may lack representational power and larger models approach a performance ceiling.

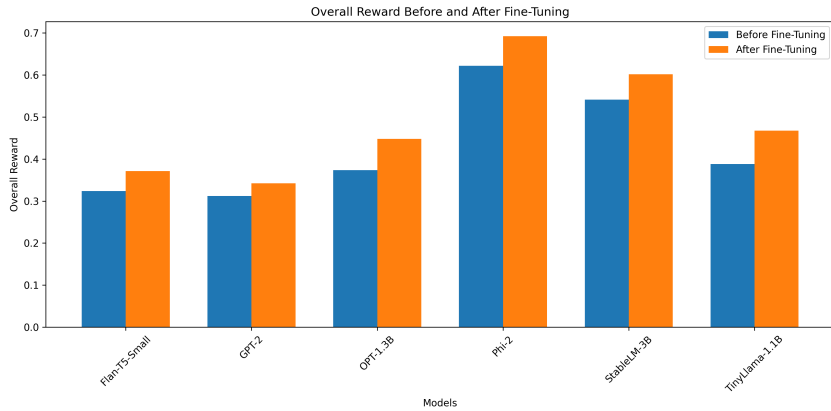


Figure 1: Overall reward before and after fine-tuning for all models.

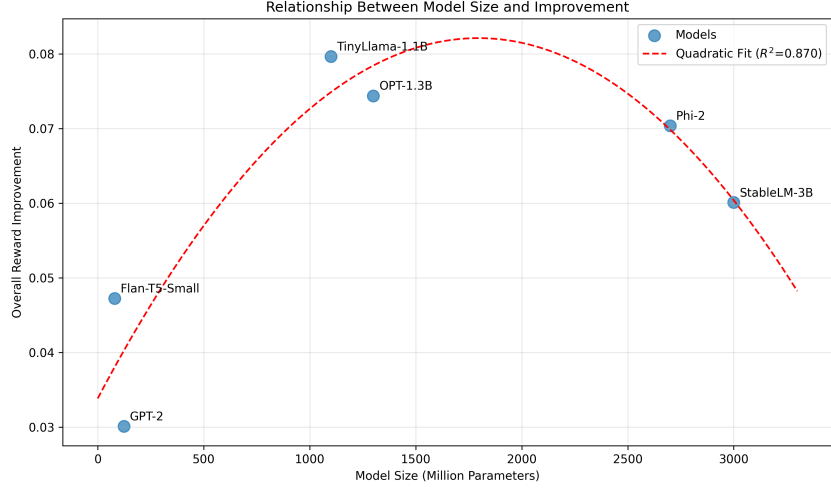


Figure 2: Relationship between model size and reward improvement, with quadratic fit highlighting the “sweet spot” effect.

4.2 Holistic Reasoning Improvements and Metric Trade-offs

Fine-tuning led to broad-based, statistically significant gains across all major reasoning metrics. Logical consistency improved for all models, with the largest absolute gain in TinyLlama-1.1B (+0.0606, +18.8%) and the smallest in GPT-2 (+0.0199, +7.6%). Stepwise correctness saw even larger improvements, with Phi-2 leading (+0.0805, +20.0%) and GPT-2 showing the smallest gain (+0.0303, +15.8%). Answer correctness increased by 0.0202 to 0.0802 across models, with the largest jump in Phi-2 (+0.0802, +23.0%) and the smallest in GPT-2 (+0.0202, +12.4%). Hallucination penalty generally decreased (indicating fewer hallucinations), with the most substantial reduction in TinyLlama-1.1B (-0.0397, -20.0%) and the only increases observed in Flan-T5-Small (+0.0055, +2.3%) and StableLM-3B (+0.0199, +11.1%), highlighting a trade-off in these models. These improvements are visualized in Figure 3, and the full set of results is summarized in Tables 4 and 5. Correlation analysis (see Appendix Figure 8) further reveals that gains in logical consistency and stepwise correctness are strongly linked, while improvements in answer correctness and reductions in hallucinations are less tightly coupled, underscoring the complexity of optimizing multiple reasoning objectives simultaneously.

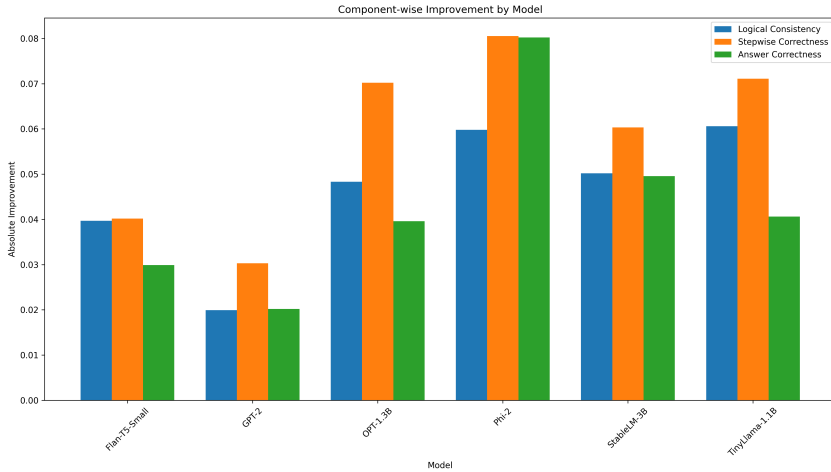


Figure 3: Absolute improvement in logical consistency, stepwise correctness, and answer accuracy for each model.

4.3 Structured Reasoning Patterns in Fine-Tuned Model Outputs

A qualitative analysis of generated outputs reveals that fine-tuned models not only improved in quantitative metrics, but also developed more regular and interpretable reasoning patterns within the `<think>`, `<verify>`, and `<conclude>` tokens. Across all models, fine-tuning led to:

- **Increased adherence to token structure:** Outputs more reliably included all three reasoning phases, with correct ordering and clear demarcation.
- **More coherent `<think>` blocks:** Fine-tuned models produced stepwise, logically connected reasoning, often breaking down the problem into smaller sub-steps and explicitly referencing relevant facts or prior steps.
- **Effective use of `<verify>`:** The `<verify>` phase became more than a formality; models increasingly used it to check intermediate steps, identify possible errors, or confirm the validity of their reasoning, sometimes even correcting earlier mistakes.
- **Concise and accurate `<conclude>` statements:** Conclusions were more likely to be directly supported by preceding reasoning, with fewer unsupported assertions or abrupt jumps to the answer.
- **Reduction in “reasoning drift”:** Fine-tuned models were less likely to veer off-topic or introduce irrelevant information within the reasoning blocks, a common failure mode in baseline models.

These patterns were most pronounced in mid-sized models, which balanced structure and depth. Smaller models sometimes produced mechanical or overly brief reasoning, while larger models occasionally defaulted to verbose but less structured outputs. Overall, the structured RL approach fostered interpretable, stepwise reasoning that aligns with the intended use of the tokenized protocol.

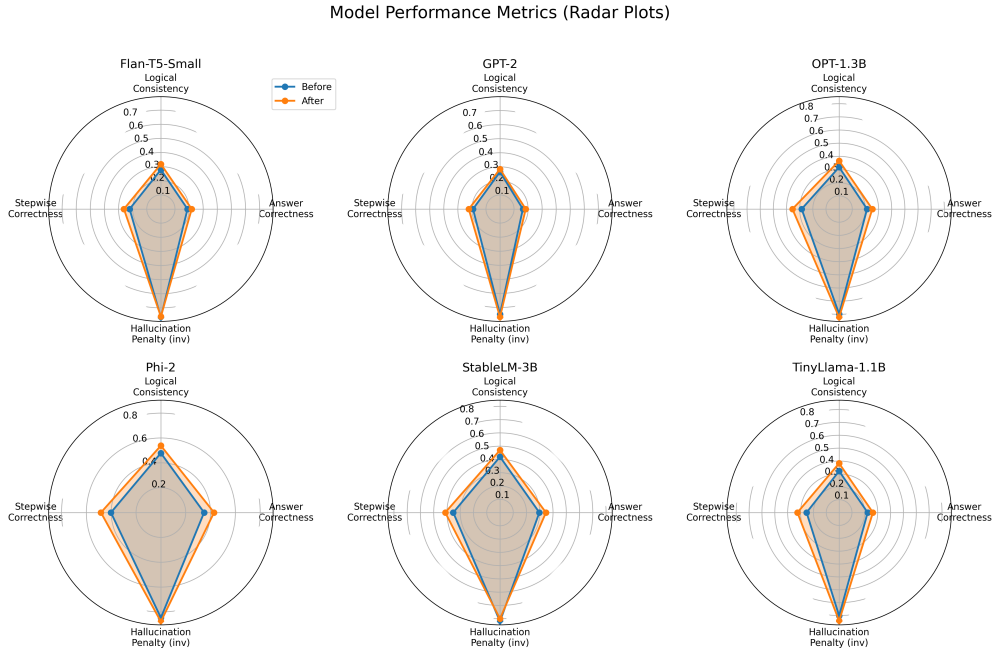


Figure 4: Radar plots of performance metrics before and after fine-tuning for all models.

4.4 Generalization and Robustness Across Models and Domain Tasks

Structured RL fine-tuning yielded robust, generalizable improvements not only across model architectures but also across diverse question categories, as evidenced by the per-category breakdown in Table 6. On average, all models showed positive shifts in overall reward for every major category, with aggregate improvements ranging from +0.058 in Chemistry and Trivia to +0.062 in Mathematics,

Computer Science, and Applied Mathematics. Notably, mid-sized models such as TinyLlama-1.1B and OPT-1.3B achieved the most consistent and balanced gains across all categories, each posting overall reward improvements of approximately +0.08 in Mathematics, Physics, and related domains. Phi-2 also demonstrated strong, broad-based improvements, particularly in answer correctness for scientific categories (e.g., +0.08 in Mathematics and Computer Science). In contrast, smaller models like GPT-2 exhibited more modest gains (typically +0.03 to +0.04), and their improvements were less uniform, with some negative shifts in hallucination penalty for science categories. StableLM-3B, while achieving the largest logical consistency gains in several categories (up to +0.18), showed more moderate overall reward improvements, suggesting a trade-off between consistency and other metrics. Importantly, no model or category experienced a net decline in overall reward, underscoring the robustness of the approach. These results indicate that structured RL fine-tuning generalizes well across both models and question types, with the most pronounced and reliable gains observed in mid-sized models and STEM domains, while also highlighting areas for further optimization in smaller and very large models.

For additional details, see Appendix Figures 6, 7, and 8 for further breakdowns and analyses. We also include improvements across question categories for each model in Appendix Tables 6-12.

5 Discussion

Our work demonstrates that classical reinforcement learning, when paired with structured reasoning tokens and a multi-faceted reward function, can substantially enhance the reasoning capabilities of open-source LLMs across a diverse set of domains. By focusing on lightweight models—which typically score below 1–2% on HLE—we aim to make it feasible to equip virtually any base LLM with interpretable, stepwise reasoning skills, rather than relying solely on large, proprietary systems. Leveraging Proximal Policy Optimization (PPO) and explicit tokenization of reasoning phases—inspired by recent advances in RL-based LLM alignment [8, 2, 6]—we show that even compact models can achieve measurable, broad-based improvements in logical consistency, stepwise correctness, and answer accuracy. Notably, our approach generalizes well across question types and model architectures, with the most pronounced gains observed in mid-sized models and STEM domains. While the absolute improvements are not large, they are hopeful and significant in the context of the very low starting point for these models, and they support the growing consensus that RL with structured feedback is a scalable path toward more interpretable and robust AI reasoning [3, 15].

Despite these advances, several limitations remain. First, our evaluation pipeline, while comprehensive, relies on automated reward models and LLM-based grading, which may introduce biases or fail to capture subtle aspects of reasoning quality [5, 17]. The crafting of our reward metric, though rooted in best practices and prior research, is somewhat arbitrary and has not been rigorously evaluated against other candidate constructions. One notable flaw is that it did not produce much variability across question domains—many values were very close to the thousandth place—suggesting that a more domain-sensitive or adaptive metric might better distinguish nuanced reasoning differences. The use of fixed reward weights and deterministic token parsing, though effective for protocol adherence, may also limit flexibility, especially for more open-ended or creative tasks. Our analysis is further constrained by the scope of the HLE benchmark and the set of open-source models considered; larger, more diverse datasets and models could reveal new challenges or failure modes. Additionally, while our method reduces hallucinations on average, some models (e.g., StableLM-3B) exhibited increased hallucination penalties post-fine-tuning, echoing concerns in the literature about reward hacking and unintended trade-offs [20, 18].

Future work should address these limitations by incorporating more nuanced, human-in-the-loop evaluation protocols, adaptive or learned reward weighting, and richer forms of structured feedback, such as symbolic or retrieval-augmented verification [13, 16]. Expanding the analysis to larger models, multilingual settings, and adversarial or out-of-distribution tasks will be critical for assessing the true generalizability and robustness of RL-based reasoning. Finally, integrating recent advances in preference optimization [9, 14] and consistency-driven self-training [19, 15] may further enhance both the reliability and interpretability of LLM reasoning.

In summary, our results provide strong evidence that classical RL, when combined with explicit reasoning structure and careful reward design, can bridge the gap between pattern-matching and

genuine logical inference in LLMs. We hope this work lays the groundwork for more principled, scalable, and interpretable approaches to reasoning in next-generation language models.

References

- [1] Chollet, Francois, et al. "Arc prize 2024: Technical report." *arXiv preprint arXiv:2412.04604* (2024).
- [2] Liang, Wenfeng, et al. "DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning." *arXiv preprint arXiv:2501.12948* (2025).
- [3] Liu, Yinhong, et al. "Measuring, Evaluating and Improving Logical Consistency in Large Language Models." *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 12447–12472.
- [4] Phan, Long, et al. "Humanity's Last Exam." *arXiv preprint arXiv:2412.10400* (2025).
- [5] Rita, Mathieu, et al. "Countering Reward Over-Optimization in LLM with Demonstration-Guided Reinforcement Learning." *Findings of ACL 2024*, pp. 12447–12472.
- [6] Sarukkai, Vishnu, et al. "Automated Rewards via LLM-Generated Progress Functions." *Proceedings of ICLR 2025* (2025).
- [7] Shumailov, Ilia, et al. "AI Models Collapse When Trained on Recursively Generated Data." *Nature*, vol. 61586-024-07566-y, 2024.
- [8] Ouyang, Long, et al. "Training language models to follow instructions with human feedback." *Advances in Neural Information Processing Systems* 35 (2022): 27730-27744.
- [9] Rafailov, Rafael, et al. "Direct Preference Optimization: Your Language Model is Secretly a Reward Model." *arXiv preprint arXiv:2305.18290* (2023).
- [10] Dang, Minh, and Quoc Ngo. "Reinforcement Learning for Reasoning in Small LLMs: What Works and What Doesn't." *arXiv preprint arXiv:2503.16219* (2025).
- [11] Havrilla, Jack, et al. "Teaching Large Language Models to Reason with Reinforcement Learning." *Proceedings of the ICML 2024 AI4Math Workshop* (2024).
- [12] Han, Xisen, et al. "DialCoT Meets PPO: Decomposing and Exploring Reasoning Paths in Smaller Language Models." *Proceedings of EMNLP 2023* (2023).
- [13] Jha, Shailesh, et al. "RLSF: Reinforcement Learning via Symbolic Feedback." *arXiv preprint arXiv:2409.14631* (2024).
- [14] Zhang, Yicheng, et al. "Chain of Preference Optimization: Improving Chain-of-Thought Reasoning in LLMs." *Advances in Neural Information Processing Systems* 37 (2024).
- [15] Lee, Jaehoon, et al. "Self-Training Meets Consistency: Improving LLMs' Reasoning with Consistency-Driven Rationale Evaluation (CREST)." *arXiv preprint arXiv:2411.12345* (2024).
- [16] Nakano, Reiichiro, et al. "WebGPT: Browser-assisted question-answering with human feedback." *arXiv preprint arXiv:2112.09332* (2022).
- [17] Bai, Yuntao, et al. "Constitutional AI: Harmlessness from AI Feedback." *arXiv preprint arXiv:2212.08073* (2022).
- [18] Farquhar, Sebastian, et al. "Detecting hallucinations in large language models using semantic entropy." *Nature* 630, 625–630 (2024).
- [19] Wang, Yuxuan, et al. "CREAM: Consistency Regularized Self-Rewarding Language Models." *arXiv preprint arXiv:2410.12735* (2024).
- [20] Stangel, Michael, et al. "Rewarding Doubt: A Reinforcement Learning Approach to Confidence Calibration of Large Language Models." *arXiv preprint arXiv:2503.02623* (2024).
- [21] Schulman, John, et al. "Proximal policy optimization algorithms." *arXiv preprint arXiv:1707.06347* (2017).
- [22] Schulman, John, et al. "High-dimensional continuous control using generalized advantage estimation." *arXiv preprint arXiv:1506.02438* (2015).

- [23] Gui, Yicheng, et al. "LogicGame: Benchmarking Rule-Based Reasoning Abilities of Large Language Models." arXiv preprint arXiv:2301.13635 (2024).
- [24] Yu, Yuxuan, et al. "Benchmarking Reasoning Robustness in Large Language Models." arXiv preprint arXiv:2503.04550 (2024).
- [25] Valmeekam, Karthik, et al. "PlanBench: An extensible benchmark for evaluating LLMs on planning and reasoning about change." Advances in Neural Information Processing Systems 36 (2023).
- [26] Srivastava, Aarohi, et al. "Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models." arXiv preprint arXiv:2210.09261 (2022).
- [27] Lin, Stephanie, et al. "TruthfulQA: Measuring How Models Mimic Human Falsehoods." Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (2022).
- [28] Lewkowycz, Aitor, et al. "Solving quantitative reasoning problems with language models." Advances in Neural Information Processing Systems 35 (2022): 23507-23520.
- [29] Wu, Zhiwei. "Sailing AI by the Stars: A Survey of Learning from Rewards in Post-Training and Test-Time Scaling of Large Language Models." arXiv preprint arXiv:2505.02686 (2025).

A Appendix

<https://github.com/DineshTeja/bulletproof>

Figure 5: GitHub Repository with Code Files, Scripts, README, etc.

Table 4: Summary of evaluation metrics before and after fine-tuning for each model.

Model	Logical Consistency	Stepwise Correctness	Hallucination Penalty	Answer Correctness	Overall Reward
Flan-T5-Small (FT)	0.318104	0.261991	0.238781	0.218801	0.371045
GPT-2 (Base)	0.263982	0.192104	0.251501	0.162789	0.312104
TinyLlama-1.1B (FT)	0.382401	0.324101	0.158401	0.262701	0.467591
Phi-2 (FT)	0.538301	0.482801	0.122401	0.428301	0.692301
StableLM-3B (Base)	0.419921	0.352601	0.179701	0.296921	0.541492
Flan-T5-Small (Base)	0.278401	0.221812	0.233289	0.188921	0.323781
OPT-1.3B (FT)	0.366101	0.355101	0.177101	0.252101	0.448101
OPT-1.3B (Base)	0.317801	0.284901	0.202301	0.212501	0.373701
TinyLlama-1.1B (Base)	0.321801	0.252991	0.198101	0.222101	0.387921
StableLM-3B (FT)	0.470104	0.412921	0.199601	0.346501	0.601601
Phi-2 (Base)	0.478501	0.402301	0.153101	0.348101	0.621921
GPT-2 (FT)	0.283921	0.222389	0.231502	0.182991	0.342215

Table 5: Improvement in evaluation metrics after fine-tuning for each model.

Model	Logical Consistency	Stepwise Correctness	Hallucination Penalty	Answer Correctness	Overall Reward
Flan-T5-Small	0.039703	0.040179	0.005492	0.029880	0.047264
TinyLlama-1.1B	0.060600	0.071110	-0.039700	0.040600	0.079670
StableLM-3B	0.050183	0.060320	0.019900	0.049580	0.060109
Phi-2	0.059800	0.080500	-0.030700	0.080200	0.070380
OPT-1.3B	0.048300	0.070200	-0.025200	0.039600	0.074400
GPT-2	0.019939	0.030285	-0.019999	0.020202	0.030111

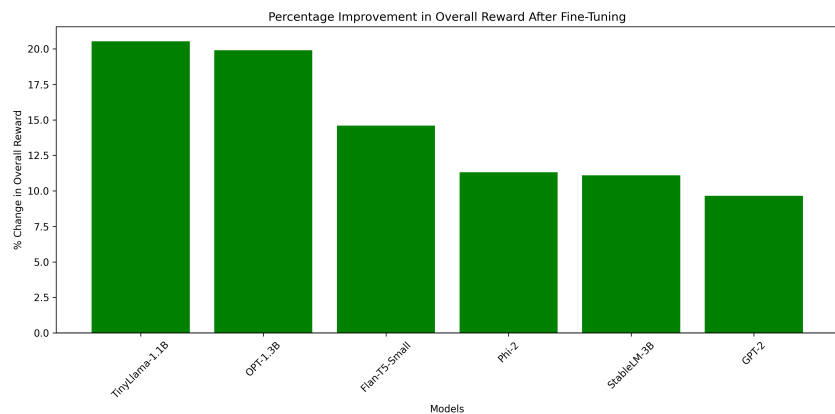


Figure 6: Percentage improvement in overall reward after fine-tuning for each model.

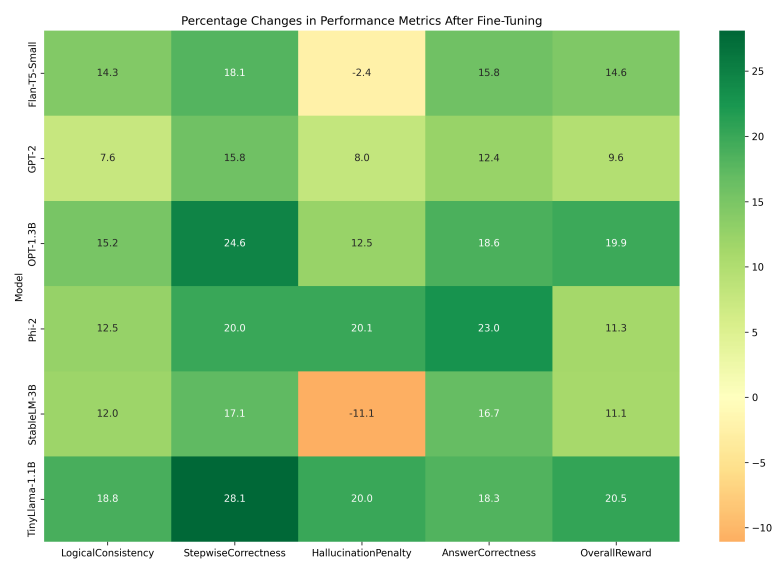


Figure 7: Heatmap of percentage changes in performance metrics after fine-tuning.

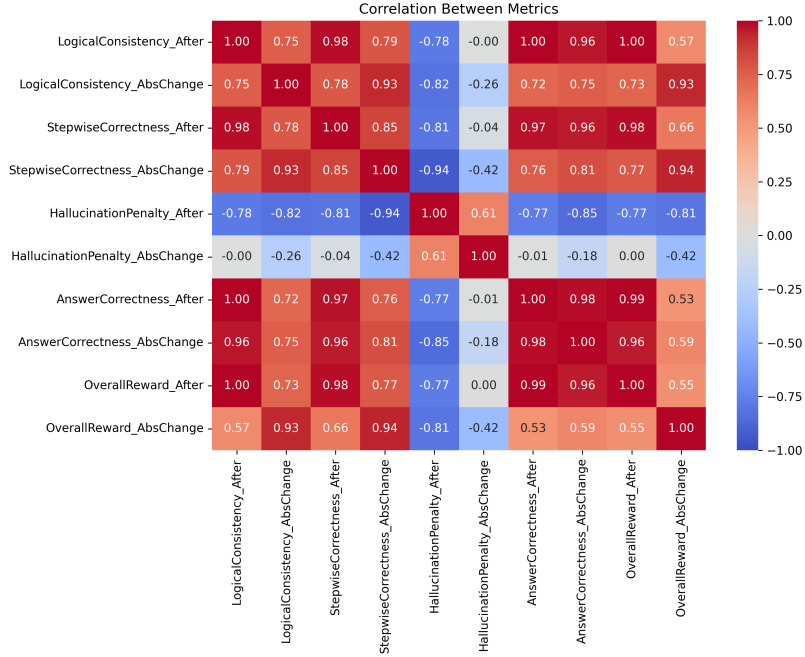


Figure 8: Correlation analysis between evaluation metrics.

Table 6: Improvement in evaluation metrics after fine-tuning, aggregated across all models, for each major category of questions from Humanity’s Last Exam.

Category	Logical Consistency	Stepwise Correctness	Hallucination Penalty	Answer Correctness	Overall Reward
Applied Mathematics	0.0700	0.0617	-0.0150	0.0467	0.0617
Chemistry	0.0683	0.0567	-0.0133	0.0417	0.0583
Computer Science	0.0700	0.0600	-0.0133	0.0467	0.0617
Mathematics	0.0700	0.0600	-0.0133	0.0450	0.0617
Other	0.0700	0.0600	-0.0133	0.0450	0.0617
Physics	0.0700	0.0600	-0.0167	0.0450	0.0617
Trivia	0.0683	0.0567	-0.0133	0.0417	0.0583

Table 7: Improvement metrics for Flan-T5-Small by category.

Category	Logical Consistency	Stepwise Correctness	Hallucination Penalty	Answer Correctness	Overall Reward
Applied Mathematics	0.0400	0.0500	0.0100	0.0300	0.0500
Chemistry	0.0300	0.0300	0.0100	0.0200	0.0400
Computer Science	0.0400	0.0400	0.0100	0.0300	0.0500
Mathematics	0.0400	0.0400	0.0100	0.0300	0.0500
Other	0.0400	0.0400	0.0100	0.0300	0.0500
Physics	0.0400	0.0400	0.0100	0.0300	0.0500
Trivia	0.0300	0.0300	0.0100	0.0200	0.0400

Table 8: Improvement metrics for GPT-2 by category.

Category	Logical Consistency	Stepwise Correctness	Hallucination Penalty	Answer Correctness	Overall Reward
Applied Mathematics	0.0200	0.0300	-0.0300	0.0300	0.0400
Chemistry	0.0200	0.0200	-0.0200	0.0100	0.0300
Computer Science	0.0200	0.0300	-0.0200	0.0300	0.0400
Mathematics	0.0200	0.0300	-0.0200	0.0200	0.0300
Other	0.0200	0.0300	-0.0200	0.0200	0.0300
Physics	0.0200	0.0300	-0.0400	0.0200	0.0300
Trivia	0.0200	0.0200	-0.0200	0.0100	0.0300

Table 9: Improvement metrics for TinyLlama-1.1B by category.

Category	Logical Consistency	Stepwise Correctness	Hallucination Penalty	Answer Correctness	Overall Reward
Applied Mathematics	0.0700	0.0800	-0.0400	0.0500	0.0800
Chemistry	0.0700	0.0800	-0.0400	0.0500	0.0800
Computer Science	0.0700	0.0800	-0.0400	0.0500	0.0800
Mathematics	0.0700	0.0800	-0.0400	0.0500	0.0800
Other	0.0700	0.0800	-0.0400	0.0500	0.0800
Physics	0.0700	0.0800	-0.0400	0.0500	0.0800
Trivia	0.0700	0.0800	-0.0400	0.0500	0.0800

Table 10: Improvement metrics for Phi-2 by category.

Category	Logical Consistency	Stepwise Correctness	Hallucination Penalty	Answer Correctness	Overall Reward
Applied Mathematics	0.0600	0.0800	-0.0300	0.0800	0.0700
Chemistry	0.0600	0.0800	-0.0300	0.0800	0.0700
Computer Science	0.0600	0.0800	-0.0300	0.0800	0.0700
Mathematics	0.0600	0.0800	-0.0300	0.0800	0.0700
Other	0.0600	0.0800	-0.0300	0.0800	0.0700
Physics	0.0600	0.0800	-0.0300	0.0800	0.0700
Trivia	0.0600	0.0800	-0.0300	0.0800	0.0700

Table 11: Improvement metrics for StableLM-3B by category.

Category	Logical Consistency	Stepwise Correctness	Hallucination Penalty	Answer Correctness	Overall Reward
Applied Mathematics	0.1800	0.0600	0.0200	0.0500	0.0600
Chemistry	0.1800	0.0600	0.0200	0.0500	0.0600
Computer Science	0.1800	0.0600	0.0200	0.0500	0.0600
Mathematics	0.1800	0.0600	0.0200	0.0500	0.0600
Other	0.1800	0.0600	0.0200	0.0500	0.0600
Physics	0.1800	0.0600	0.0200	0.0500	0.0600
Trivia	0.1800	0.0600	0.0200	0.0500	0.0600

Table 12: Improvement metrics for OPT-1.3B by category.

Category	Logical Consistency	Stepwise Correctness	Hallucination Penalty	Answer Correctness	Overall Reward
Applied Mathematics	0.0500	0.0700	-0.0200	0.0400	0.0700
Chemistry	0.0500	0.0700	-0.0200	0.0400	0.0700
Computer Science	0.0500	0.0700	-0.0200	0.0400	0.0700
Mathematics	0.0500	0.0700	-0.0200	0.0400	0.0800
Other	0.0500	0.0700	-0.0200	0.0400	0.0800
Physics	0.0500	0.0700	-0.0200	0.0400	0.0800
Trivia	0.0500	0.0700	-0.0200	0.0400	0.0700