Grounded Chess Reasoning in Language Models via Master Distillation

Zhenwei Tang[1], **Qianfeng Wen**[1], **Seth Grief-Albert**[2], **Yahya Elgabra**[1], **Blair Yang**[3], **Honghua Dong**[1] **& Ashton Anderson**[1]

[1]Department of Computer Science, University of Toronto
[2]Queen's University [3]Coolwei AI Lab
Contact: {josephtang, ashton}@cs.toronto.edu

Code 🤗 Dataset 🤗 Model

Abstract

Language models often lack grounded reasoning capabilities in specialized domains where training data is scarce but bespoke systems excel. We introduce a general framework for distilling expert system reasoning into natural language chain-of-thought explanations, enabling compact models to acquire domain expertise and the ability to generate faithful, grounded explanations. Rather than distilling only final outputs, we capture the full reasoning process, transforming opaque expert computations into transparent, step-by-step explanations. We demonstrate this approach in chess, a canonical reasoning domain where language models continue to underperform. Our 4B parameter model, C1, advances from a near-zero baseline to 48.1% accuracy, outperforming all open-source models and most frontier proprietary systems. Notably, C1 surpasses its distillation teacher and generates solutions in two orders of magnitude fewer tokens than baselines. Unlike prior neural chess approaches that predict only best moves, C1 generates explainable solutions revealing strategic reasoning. Our pipeline combines supervised fine-tuning and reinforcement learning with theme-balanced data sampling for comprehensive tactical coverage. Master Distillation demonstrates how to inject expert-level knowledge into compact models for under-optimized domains, offering a recipe for unlocking RLVR where LLMs lack sufficient base capabilities.

1 Introduction

Large language models exhibit jagged intelligence: superhuman performance in some domains while remaining surprisingly clumsy in others. While the reasons are multifaceted, weak performance often emerges in niche domains where high-quality reasoning data is sparse in pretraining corpora, even when strong specialized systems exist. Chess exemplifies this phenomenon: despite the existence of superhuman engines like Stockfish (Stockfish Team, 2023), state-of-the-art LLMs struggle with basic tactical puzzles, and recent attempts to apply reinforcement learning with verifiable rewards (RLVR) have failed to yield meaningful improvements (Hwang et al., 2025a).

Prior work has explored distilling chess engines into neural networks, but these approaches transfer only move predictions without the underlying reasoning process (Ruoss et al., 2024). We present Master Distillation, a data synthesis approach that distills both outcome and process from specialized systems into language models. The core insight is that expert systems know the correct answer but cannot articulate their reasoning, while LLMs can generate fluent explanations but lack domain knowledge. Master Distillation combines these complementary strengths: we leverage a master system to provide ground-truth solutions and prompt a frontier LLM to verbalize the reasoning process through Feigned Discovery Prompting, generating chain-of-thought traces that are both verifiably correct and pedagogically natural.

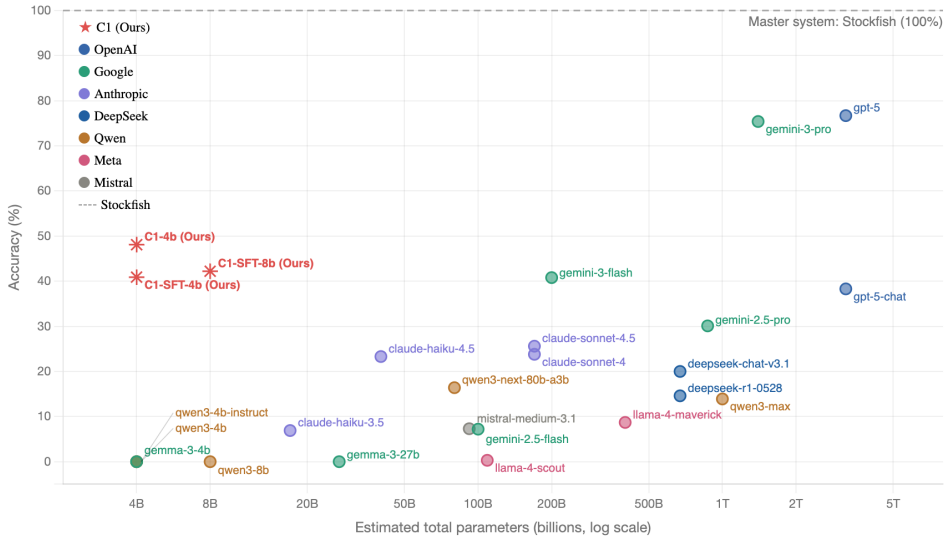


Figure 1: C1 achieves competitive performance against frontier models orders of magnitude larger on chess puzzle-solving.

We focus on chess puzzle solving as our testbed. Chess offers perfectly verifiable rewards through engine evaluation, and puzzles have an unambiguous ground truth with a single best move, making them ideal for both evaluation and reward design. The domain is rich with tactical concepts spanning multiple difficulty levels, enabling systematic analysis of model capabilities. Most importantly, the failure of prior RLVR attempts on chess (Hwang et al., 2025a) makes it a compelling challenge: if Master Distillation can unlock RLVR here, the approach likely generalizes to other domains where LLMs currently fall short while master systems exist.

Our experiments demonstrate that Master Distillation enables effective SFT+RLVR training for chess reasoning. Our C1-4B model achieves 48.1% accuracy, outperforming all open-source models and most proprietary models. Notably, C1-4B surpasses its distillation teacher Gemini-3-Flash (40.8%), demonstrating that the student can exceed teacher performance. Beyond accuracy, our models exhibit remarkable token efficiency, generating solutions in roughly two orders of magnitude fewer tokens than both proprietary and open-source baselines, reflecting how humans actually solve chess puzzles through focused pattern recognition rather than extensive deliberation. In summary, we show that Master Distillation unlocks RLVR for domains where LLMs lack sufficient base capabilities, offering a recipe for improving language model performance in specialized domains with strong bespoke systems.

2 Related Work

Reinforcement Learning with Verifiable Rewards. RLVR enhances language model reasoning by replacing learned reward models with deterministic verification Guo et al. (2025); Hwang et al. (2025b); Shao et al. (2024); Yu et al. (2025), providing objective signals that scale efficiently unlike preference-based RLHF Ouyang et al. (2022). The success of recent reasoning models, including OpenAI o1 OpenAI (2024), DeepSeek-R1 Guo et al. (2025), and Qwen3 Yang et al. (2025a), demonstrates the effectiveness of RLVR. These methods build on policy gradient foundations like PPO Schulman et al. (2017), with GRPO Shao et al. (2024) marking a critical pivot by eliminating critic networks via grouped rollout baselines. Subsequent work addresses GRPO's limitations: DAPO Yu et al. (2025) introduces clip-higher bounds and dynamic sampling, GSPO Zheng et al. (2025) stabilizes long sequences via sequence-level ratios, Dr. GRPO Liu et al. (2025) corrects length biases, PRIME Cui et al. (2025) enables

implicit process rewards, and RLOO Ahmadian et al. (2024) revisits REINFORCE-style optimization. RLVR has not yet been shown to be effective in chess Hwang et al. (2025b).

Chess Move Modeling. Traditional chess AI focuses on finding optimal moves through search and evaluation. AlphaZero (Silver et al., 2016; 2017a;b) and Leela Chess Zero (Pascutto & Linscott, 2018; Pascutto et al., 2018) combine Monte Carlo tree search with neural network evaluation, achieving superhuman performance. Stockfish (Stockfish Team, 2023) integrates Efficiently Updateable Neural Networks (NNUE) (Nasu, 2018) for fast position evaluation. Recent work explores replacing search with pure neural prediction: Ruoss et al. (2024) distill search-based policies into transformers, and Monroe & Chalmers (2024) train transformers directly on grandmaster games. These approaches focus on move prediction but do not produce human-readable explanations of their reasoning. A parallel line of research models human chess behavior rather than optimal play. Maia (McIlroy-Young et al., 2020) predicts population-level human moves, extended by Maia-2 (Tang et al., 2024) and ALLIE (Zhang et al., 2025a). Individual-level modeling has been explored in Maia-Individual (McIlroy-Young et al., 2022) and Maia4All (Tang et al., 2026). While these models capture human decision-making patterns, they similarly lack explicit reasoning traces.

Chess LLMs. Several benchmarks evaluate LLM chess capabilities across different dimensions: MATE (Wang et al., 2025), PGN2FEN (Cooper, 2025), LLM Chess Puzzles (Kagi Search, 2025), LLM Chess Leaderboard (Saplin, 2025), Chess LLM Arena (Jannala, 2025), Kaggle Game Arena (Lee et al., 2025), and ChessQA (Wen et al., 2025). Kim et al. (2025) explores concept-guided chess commentary generation. These benchmarks reveal that current LLMs struggle with chess reasoning despite strong performance in other domains. Early efforts in training LLMs for chess include commentary generation (Jhamtani et al., 2018; Zang et al., 2019), ChessGPT (Feng et al., 2023), and self-supervised approaches using GPT-2 (Noever et al., 2020). Stöckl (2021) analyze language model learning dynamics on chess, while ChePT (Swingle et al., 2021) combines move prediction with commentary generation. Zhang et al. (2025b) train on complete game transcripts to improve move prediction. However, none of these approaches explicitly address distilling reasoning from master systems like chess engines into language models for explainable puzzle solving, which is the focus of our work. Most recently, Chess-R1 (Hwang et al., 2025a) attempted to apply RLVR directly to chess but found that neither direct RLVR nor SFT followed by RLVR yields meaningful improvements. Our work addresses this challenge through Master Distillation, demonstrating that the SFT+RLVR pipeline can be made effective for chess reasoning with a properly designed distillation method.

3 Methodology

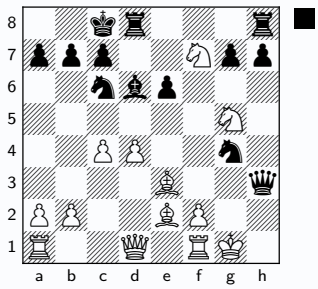
3.1 Chess Puzzle Solving with RLVR

Chess Puzzles. Chess puzzles are carefully curated tactical positions with a single objectively best move sequence, designed to test concrete calculation and pattern recognition across various tactical themes such as forks, pins, and mating attacks. Compared to arbitrary game positions where multiple reasonable moves may exist, puzzles offer unambiguous ground truths, making them particularly suitable for evaluation and reward design in our setting.

Chess Puzzle Solving as a Distillation Task. We formulate chess puzzle solving as predicting the correct first move. While this may appear simple, finding the optimal first move requires the model to reason about future positions, anticipate opponent responses, and evaluate resulting outcomes before committing to a decision. This lookahead reasoning is precisely what distinguishes tactical calculation from superficial pattern matching. Prior work on distilling chess engines into neural networks (Ruoss et al., 2024; Nasu, 2018) focuses on transferring the engine's move predictions or position evaluations, but not the underlying reasoning process. In contrast, we focus on distilling both the outcome and the process: the model generates a chain-of-thought that explores candidate moves and evaluates their consequences before outputting a final answer. The model is evaluated on

first-move accuracy, which serves as a proxy for the quality of its internal reasoning learned from distillation.

Example Puzzle



Black To Move. Board shown for reference only, not included in prompt.

```
You are given a chess position in FEN:
2kr3r/ppp2Npp/2nbp3/6N1/2PP2n1/4B2q/PP2BP2/R2Q1RK1
b - - 2 15. Find the best move for the side to
play. PIECE_ARRANGEMENT. LEGAL_MOVES. Analyze step
by step and explain your reasoning. Finish with
a single line formatted EXACTLY as: FINAL_ANSWER:
<answer>. Use UCI notation (e.g., e2e4, c2b1q)
for the final answer. Ground Truth: h3h2
```

Established Post Training Pipeline. Reinforcement learning with verifiable rewards (RLVR), with optional supervised fine-tuning (SFT) as a cold-start phase, has emerged as the dominant paradigm for enhancing LLM reasoning capabilities. This pipeline has proven effective across diverse domains: in mathematics, coding, deep research, and tool use Guo et al. (2025); Yang et al. (2025b); Jin et al. (2025); Li et al. (2025). The success of RLVR relies on a critical prerequisite: the base model must already possess sufficient capability to occasionally produce correct solutions, which RL can then amplify through reward-driven optimization. Chess, however, presents a unique challenge. Despite being a domain with perfectly verifiable rewards, neither direct RLVR nor SFT followed by RLVR yields meaningful improvements (Hwang et al., 2025a). This is because LLMs' base chess capabilities remain too weak to bootstrap from, stemming from the relative scarcity of high-quality chess reasoning data in pretraining corpora. Unlike mathematics, where step-by-step solutions are abundant online, chess analysis rarely includes the detailed chain-of-thought reasoning needed for effective SFT.

3.2 Master Distillation

We present a data synthesis approach that combines the domain expertise of specialized systems, e.g., the master, with the linguistic capabilities of frontier LLMs. The core insight is that bespoke systems like Stockfish (Stockfish Team, 2023) know the correct answer but cannot articulate their symbolic reasoning, while LLMs can generate fluent explanations but lack the domain knowledge to reliably solve hard problems. This motivates our Master Distillation approach: by leveraging a specialized master system to provide ground-truth solutions and a frontier LLM to verbalize the symbolic reasoning process, we can generate high-quality training data that bridges this gap and enables effective RLVR for chess reasoning.

3.2.1 Context Engineering

The first dimension that requires careful design towards realizing is context engineering: we must provide the LLM with sufficient information to understand the position and generate grounded analysis.

Master Solution. The principal variation (PV) is the sequence of optimal moves for both sides leading to the puzzle's objective, which may be checkmate, material gain, or achieving a decisive advantage. The PV represents the symbolic reasoning process of the master system, and is the core knowledge we aim to distill from the symbolic master into the language model student. We leverage Stockfish (Stockfish Team, 2023) at depth 24, which is more than capable of solving most puzzles accurately, as the master system to compute the PV for each puzzle position. We choose not to use transformer-based chess models (Pascutto

& Linscott, 2018; Ruoss et al., 2024) because they do not explicitly provide PVs or allow trivial extraction of move sequences. We consider two settings: *multi-PV*, which generates the top-k candidate move sequences and provides additional context about alternative move options, and *best-move PV*, which provides only the single optimal line. While multi-PV offers richer information, best-move PV introduces less noise, allowing the model to focus on what is correct. This is particularly suitable for puzzles where there is one absolute best move and alternatives are significantly worse. We provide empirical comparison between these settings in Table 2.

Board and Move Representation. Chess positions are encoded in Forsyth-Edwards Notation (FEN), a compact string representation that captures piece placement, active color, castling rights, en passant targets, and move clocks (Edwards, 1994). Moves are represented in Universal Chess Interface (UCI) notation, which specifies moves by their source and destination squares (e.g., e2e4). While Standard Algebraic Notation (SAN) is more human-readable and prevalent in online chess discussions and literature, it introduces ambiguities since the same symbol can refer to different pieces (e.g., B for bishop versus the b-file) and includes special characters like + and # that complicate evaluation and reward computation through inconsistent string matching. Prior work (Tang et al.) has shown that LLMs struggle to recognize FEN strings due to tokenization issues, conflating chess reasoning ability with position recognition. To separate these concerns and provide complete positional context, we augment the FEN with an explicit piece list describing the location of each piece and a list of all legal moves in the current position, for both the puzzle-solving task and distillation. An example chess puzzle is shown in the blue box titled "Example Puzzle".

Additional Hints. Puzzles may be associated with additional metadata, as provided in public databases like Lichess[1]. We provide the teacher LLM with additional context to guide its analysis. The *opponent's last move* provides crucial context for understanding the tactical situation, as many puzzles arise from the opponent's mistake or create threats that must be addressed. The *tactical themes* (e.g., fork, pin, back-rank mate) hint at the underlying pattern the LLM should explain, helping it focus on the relevant tactical motifs rather than generic analysis. The *puzzle rating* signals the expected complexity, allowing the LLM to calibrate the depth and length of its explanation accordingly. Crucially, while all this information guides the teacher's generation, the student model never sees these hints during training or evaluation.

3.2.2 Feigned Discovery Prompting

Our goal is to produce chain-of-thought traces that are both verifiably correct, grounded in the master's solution, and pedagogically natural, reading like genuine reasoning rather than post-hoc rationalization. Naively conditioning on the answer risks generating justifications for a known conclusion rather than authentic problem-solving processes. We want the student model to learn how to arrive at solutions, not merely how to justify them. This creates a fundamental tension between faithfulness to the master trace and authentic reasoning. If the generated reasoning is too faithful, the LLM simply parrots the moves without producing transferable reasoning patterns. If it is too exploratory, the LLM may hallucinate and lose the grounding benefit of the master solution. Our key insight is to instruct the teacher LLM to reason *as if* the solution is unknown, while secretly steering toward the master trace. The teacher must pretend not to know the answer, simulating the discovery process a human solver would experience. This feigned ignorance is critical: it forces the LLM to generate reasoning that explains *why* a move is correct through genuine analysis, rather than simply asserting correctness because the answer was provided. The master's PV and additional context acts as a soft constraint, ensuring the reasoning naturally converges to the correct solution while maintaining the appearance of authentic exploration.

We impose several additional constraints in the prompt to ensure the generated reasoning traces are concise and pedagogically useful. First, we enforce length constraints of 4–10 sentences scaled to puzzle complexity, as chess learners benefit from focused analysis rather than verbose explanations. Second, we require the teacher to bridge chess concepts with

[1] https://database.lichess.org/#puzzles

notation by annotating moves with natural language clarifications (e.g., "Qxh7+ (queen takes h7 with check)") and grounding analysis in explicit board coordinates (e.g., "the queen on h5", "king on g1") while explaining tactical relationships such as attacks, pins, and defender counts. This bridging is important because student LLMs have limited exposure to chess-specific notation during pretraining. Third, we enforce stylistic constraints: an objective voice without phrases like "I see" or "I notice", and strict prohibition against mentioning engine scores, ratings, themes, or any indication that the solution was provided. In essence, we are role-playing the LLM as a chess grandmaster who analyzes positions naturally and arrives at the solution through genuine reasoning. These constraints collectively ensure the generated traces read as natural expert analysis rather than machine-generated explanations.

An example of the distillation prompt is shown in the blue box titled "Master Distillation Prompt" in the Appendix.

3.3 C1 Training

Data Balancing. We apply theme-balanced sampling to construct both SFT and RLVR training data, as described in Algorithm 1 in the appendix. Since each puzzle is annotated with multiple tactical themes, we prioritize balancing by the rarest themes, ensuring that infrequent concepts such as underPromotion and balestraMate receive sufficient representation. By sampling puzzles that contain these rare themes, more frequent themes like fork and pin are naturally covered as co-occurring labels. We target M=800 samples for each of the K=50 rarest themes, resulting in approximately 40,000 samples for both SFT and RLVR with non-overlapping puzzle sets. Detailed statistics are provided in Table 7 and per-theme distributions in Table 8.

Supervised Fine-Tuning. We perform full fine-tuning on Qwen3-4B-Instruct-2507, which achieves near-zero accuracy on our evaluation out of the box. SFT is a necessary prerequisite for RLVR: the base model must solve puzzles correctly some of the time, otherwise all rollouts receive zero reward and no learning signal is available. We opt for full fine-tuning over parameter-efficient methods like LoRA Hu et al. (2022) because our task requires knowledge acquisition in a novel domain rather than adaptation of existing capabilities, which benefits from updating all model parameters.

Reinforcement Learning with Verifiable Rewards. We build upon DAPO Yu et al. (2025), which introduces improvements over GRPO Shao et al. (2024): Overlong Reward Shaping, Token-Level Policy Gradient Loss, Dynamic Sampling, Clip-Higher, and Removing KL Divergence. However, we exclude two of these modifications: Overlong Reward Shaping and Removing KL Divergence. The original DAPO removes KL divergence under the assumption that long-CoT reasoning benefits from allowing the model distribution to diverge significantly from the initial policy. In our setting, the SFT data already consists of concise, to-the-point reasoning traces, and we aim to preserve this brevity rather than encourage free exploration. Similarly, Overlong Reward Shaping is unnecessary since our responses are inherently short. We denote this modified algorithm as DAPO-C1. We use a binary reward based solely on whether the final move matches the Stockfish-verified optimal move. Following the Bitter Lesson Sutton (2019), we avoid complex reward shaping that may introduce biases or enable reward hacking, instead relying on a clean signal that scales with data and compute.

4 Experiments

4.1 Experimental Settings

We implement SFT using LLaMA-Factory (Zheng et al., 2024) and RLVR using VERL (Sheng et al., 2024). All experiments are conducted on 4×H100 GPUs. For both SFT and RLVR, we use a held-out validation set of chess puzzles for early stopping to prevent overfitting. Detailed hyperparameters for SFT and RLVR are provided in Tables 5 and 6 in the Appendix, respectively. We follow the evaluation protocol of Wen et al. (2025). The test set contains 900 puzzles: 500 puzzles sampled across 20 tactical themes (25 per theme) and 400 puzzles

Table 1: Performance comparison across difficulty levels and models. Avg Acc represents average accuracy (%), and Avg Tokens indicates average generation length.

	Beginner	Intermediate	Advanced	Expert	Theme-Split	Avg Acc	Avg Tokens
Proprietary models							
gpt-5	95.0	84.0	54.0	31.0	85.2	76.7	12,193
gemini-3-pro	88.0	86.0	70.0	44.0	78.2	75.4	3,182
gemini-3-flash	65.0	59.0	34.0	19.0	38.0	40.8	6,418
gpt-5-chat	52.0	39.0	27.0	18.0	41.8	38.3	925
gemini-2.5-pro	37.0	31.0	29.0	19.0	31.0	30.1	9,668
claude-sonnet-4.5	32.0	29.0	15.0	11.0	28.6	25.6	3,227
claude-sonnet-4	35.0	19.0	16.0	10.0	26.8	23.8	8,028
claude-haiku-4.5	33.0	24.0	14.0	11.0	25.6	23.3	8,111
gemini-2.5-flash	9.0	4.0	6.0	5.0	8.2	7.2	9,991
Open-source models							
deepseek-chat-v3.1	27.0	21.0	6.0	16.0	22.0	20.0	11,249
qwen3-next-80b-a3b	24.0	14.0	14.0	8.0	17.6	16.4	13,938
deepseek-r1-0528	11.0	10.0	14.0	16.0	16.0	14.6	14,442
qwen3-max	22.0	15.0	3.0	16.0	13.8	13.9	3,393
llama-4-maverick	12.0	8.0	5.0	10.0	8.6	8.7	1,092
mistral-medium-3.1	9.0	6.0	7.0	4.0	8.0	7.3	2,818
llama-4-scout	0.0	0.0	0.0	1.0	0.4	0.3	806
gemma-3-27b	0.0	0.0	0.0	0.0	0.0	0.0	705
Ours							
C1-SFT-4B	51.0	30.0	30.0	26.0	46.2	40.9	188
C1-SFT-8B	57.0	36.0	27.0	27.0	46.6	42.2	189
C1-4B	65.0	39.0	39.0	22.0	53.6	48.1	178

sampled across 4 difficulty levels (100 per level: Beginner, Intermediate, Advanced, Expert). We report pass@1 accuracy as the primary metric. For all baseline models, we enable reasoning mode where applicable. We use OpenRouter for API access to proprietary models during evaluation. We use Gemini-3-Flash-Preview via OpenRouter as the teacher model for Master Distillation.

4.2 Performance Comparison

Our C1-4B model achieves the highest accuracy among all open-source models by a substantial margin and outperforms most frontier proprietary models, demonstrating the effectiveness of Master Distillation for domain-specific reasoning. Notably, all C1 models surpass Gemini-3-Flash, the model used to generate these reasoning traces, indicating that combining symbolic expertise with language model reasoning enables the student to exceed its teacher. This improvement is consistent across difficulty levels, with particularly strong gains on Intermediate puzzles. On the Theme-Split evaluation, C1-4B shows even more pronounced advantages, with detailed per-theme results provided in Appendix Table 9.

Beyond accuracy, our models exhibit remarkable token efficiency, generating solutions in roughly two orders of magnitude fewer tokens than both proprietary and open-source baselines. This reflects how humans actually solve chess puzzles: through focused pattern recognition and precise calculation rather than extensive deliberation. It also aligns with the practical needs of chess learners, who benefit from concise, to-the-point explanations rather than verbose reasoning chains.

Comparing C1-4B (SFT+RLVR) with C1-SFT-4B reveals the effect of reinforcement learning. RLVR provides consistent improvements on Beginner and Intermediate puzzles, but shows diminished or negative gains on Expert-level problems. This pattern suggests that the chain-of-thought reasoning traces may not faithfully capture the underlying logic for harder puzzles, limiting what RLVR can learn from outcome-based rewards. Unlike mathematical reasoning, where RLVR often yields dramatic improvements, the less reliable reasoning chains in complex chess puzzles constrain the potential gains from reinforcement learning.

4.3 Additional Findings

SFT Data. Table 2 presents ablation studies on SFT data configurations, revealing several key insights. First, distillation quality matters: using Gemini-3-Pro as the teacher outperforms Gemini-3-Flash, suggesting that higher-quality chain-of-thought traces lead to better student

Table 2: Ablation study on SFT data configurations.

Scale	8k	8k	8k	8k	16k	8k	8k	8k	39k
Distribution	random	hard	balanced	balanced	balanced	balanced	balanced	balanced	balanced
Quality	flash	flash	pro	flash	flash	flash	flash	flash	flash
Context	full	full	full	full	full	Multi PVs	w/o Theme	w/o Feigned	full
SFT Results	19.3	16.2	22.8	20.1	29.7	17.6	17.3	16.3	40.9

Table 3: Ablation study on RLVR data and SFT configurations.

Distribution	**Reuse**	**SFT**	**RLVR**	**Impr**
random	false	40.9	47.1	+6.2
hard	false	40.9	43.0	+1.1
balanced	true	40.9	46.7	+5.8
balanced	false	40.9	48.1	+7.2

Configuration	**SFT**
LoRA ($r = 16, \alpha = 32$)	32.1
LoRA ($r = 64, \alpha = 128$)	38.8
Full Fine-Tuning	40.9

performance. However, due to budget constraints, we use Flash for scaling experiments. Second, data distribution significantly impacts results: hard sampling yields the worst performance because chain-of-thought quality degrades on difficult puzzles, while random sampling underperforms balanced sampling due to insufficient coverage of rare tactical concepts. Third, our context engineering and prompting design choices are validated: Multi-PV introduces noise and performs worse than best-move PV, removing theme hints causes the reasoning to lose focus, and removing our Feigned Discovery Prompt (w/o Feigned) leads to the largest degradation, demonstrating that our prompting strategy is critical for generating effective training data. Fourth, data scale matters substantially, with performance improving from 19.3% at 8k to 29.7% at 16k and 40.9% at 39k samples. These results collectively justify our final configuration (highlighted in gray): 39k balanced samples distilled from Gemini-3-Flash with full context and Feigned Discovery Prompting.

RLVR Data. Table 3 (left) presents ablations on RLVR data selection strategies. Random sampling draws problems i.i.d. from the training distribution, while hard sampling prioritizes the most difficult puzzles. Balanced sampling ensures equal representation across tactical themes for broader concept coverage. The results show that balanced sampling achieves the best performance, suggesting that diverse tactical exposure benefits reinforcement learning more than focusing on difficulty alone. Notably, hard sampling yields the smallest improvement, consistent with our earlier observation that unfaithful reasoning traces on difficult puzzles limit RLVR's effectiveness. We also examine whether to *reuse* SFT samples during RLVR. As shown in 3, reusing SFT samples slightly hurts performance compared to using fresh problems. This is likely because training on previously seen examples encourages the model to reinforce existing behavior rather than acquire new knowledge, shifting the learning objective toward consistency with prior outputs rather than exploration of improved reasoning strategies.

SFT Training. Table 3 (right) validates our choice of full fine-tuning over parameter-efficient methods. While increasing LoRA rank from 16 to 64 improves performance from 32.1% to 38.8%, full fine-tuning achieves the best result at 40.9%. This confirms our hypothesis that chess puzzle solving requires knowledge acquisition in a novel domain rather than adaptation of existing capabilities, benefiting from updating all model parameters rather than low-rank approximations.

RLVR Training. Table 4 (left) presents ablation studies on RLVR configurations. DAPO outperforms GRPO, validating the benefits of Token-Level Policy Gradient Loss, Dynamic Sampling, and Clip-Higher for our setting. Our modified DAPO-C1, which retains KL divergence and removes Overlong Reward Shaping to preserve concise reasoning, achieves the best performance with a 7.2% improvement over SFT. We also experiment with partial credit rewards, where moves with correct source or destination squares receive a fractional reward η. However, partial rewards with η=0.5 hurts performance, and η=0.2 yields only marginal gains. This supports our design choice of using binary correctness rewards: following the Bitter Lesson (Sutton, 2019), simple and clean reward signals scale better than hand-crafted shaping, which may introduce biases or enable reward hacking. Our final configuration (highlighted in gray) uses DAPO-C1 with binary correctness rewards.

Table 4: Ablation study on RLVR configurations and base models. Qwen3-8B was not trained with RLVR due to computational constraints. Qwen3-4B-it denotes Qwen3-4B-Instruct-2507.

Algorithm	Reward	SFT	RLVR	Impr
GRPO	Correctness	40.9	45.5	+4.6
DAPO	Correctness	40.9	47.3	+6.4
DAPO-C1	Partial (η=0.5)	40.9	39.3	-1.6
DAPO-C1	Partial (η=0.2)	40.9	41.6	+0.7
DAPO-C1	Correctness	40.9	48.1	+7.2

Model	SFT	RLVR
Qwen3-8B	42.2	-
Qwen3-4B	40.5	47.6
Qwen3-4B-it	40.9	48.1
Gemma3-4B	36.2	41.0

Model. Table 4 (right) compares different base models after SFT and RLVR training. Qwen3-4B-Instruct-2507 achieves the best overall performance, slightly outperforming the reasoning variant Qwen3-4B, possibly because reasoning models are more rigid in their established reasoning patterns and thus harder to adapt to domain-specific chain-of-thought styles. Gemma3-4B lags behind the Qwen models, reflecting differences in pretraining data and architecture. Notably, the 8B model shows only marginal gains over the 4B model after SFT (42.2% vs 40.9%), indicating that data quality is a more significant bottleneck than model capacity at this scale. This suggests that simply scaling parameters without improving training data yields diminishing returns for domain-specific reasoning tasks. While scaling to larger models would likely yield further improvements, we focus on the 4B model due to computational constraints and leave scaling experiments to future work.

5 Discussion

Towards Better Performance. Our ablation studies reveal several key factors for effective chess reasoning: high-quality chain-of-thought traces from capable teacher models, balanced data distribution for broad tactical coverage, careful context engineering and Feigned Discovery Prompting, and sufficient data scale. Each of these dimensions offers opportunities for improvement. The most straightforward path to better performance is scaling. Our experiments show consistent gains from 8k to 16k to 39k training samples, with no sign of saturation. Similarly, using Gemini-3-Pro instead of Flash as the teacher improves performance, suggesting that even stronger teachers would yield higher-quality distillation data. Notably, our distilled C1 models surpass their teacher Gemini-3-Flash, demonstrating that Master Distillation combined with RLVR can produce students that exceed teacher performance. Due to resource constraints, we focused on 4B parameter models, but our methodology naturally extends to larger architectures where we would expect further gains.

Chess Foundational Model. While C1 performs well on puzzles with unambiguous ground truth, a true chess foundation model must generalize to strategic positions where multiple reasonable continuations exist and evaluation is less binary. This requires rethinking reward design for Master Distillation. Beyond move prediction, such a model could support game annotation, opening preparation, endgame analysis, mistake identification, and personalized training. Mid-training on large-scale chess corpora with multimodal capabilities to process board images could further enhance utility. The ultimate goal is accessible chess education: unlike engines that provide optimal but opaque suggestions beyond human comprehension, a foundation model trained with Master Distillation could explain concepts at appropriate skill levels and guide learners through tactical and strategic themes, democratizing high-quality chess instruction.

Master Distillation Generalization. Master Distillation applies to any domain where strong bespoke systems exist but LLMs lack reasoning capabilities due to sparse training data. The key requirements are a master system providing verifiable solutions and problems with clear correctness criteria. Examples include circuit design (simulation-validated), music composition (harmonic/counterpoint verification), and molecular docking (binding affinity scoring), all domains where automated verification exists but LLM-generated reasoning remains limited. While the master system component transfers directly, Feigned Discovery Prompting requires domain-specific redesign: the core principle of the LLM pretending to discover the solution through genuine reasoning remains constant, but what constitutes authentic reasoning varies by domain.

References

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms, 2024. URL https://arxiv.org/abs/2402.14740.

Aidan Cooper. Pgn2fen: A benchmark for evaluating llm chess reasoning. Blog post, May 2025. URL https://www.aidancooper.co.uk/pgn2fen-benchmark/. Accessed 2025-09-16.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, Jiarui Yuan, Huayu Chen, Kaiyan Zhang, Xingtai Lv, Shuo Wang, Yuan Yao, Xu Han, Hao Peng, Yu Cheng, Zhiyuan Liu, Maosong Sun, Bowen Zhou, and Ning Ding. Process reinforcement through implicit rewards, 2025. URL https://arxiv.org/abs/2502.01456.

Steven J Edwards. Portable game notation specification and implementation guide, 1994. Chess programming standard.

Xidong Feng, Yicheng Luo, Ziyan Wang, Hongrui Tang, Mengyue Yang, Kun Shao, David Mguni, Yali Du, and Jun Wang. Chessgpt: Bridging policy learning and language modeling. *Advances in Neural Information Processing Systems*, 36:7216–7262, 2023.

Daya Guo, Dejian Yang, and Haowei et al Zhang. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, September 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL http://dx.doi.org/10.1038/s41586-025-09422-z.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

Dongyoon Hwang, Hojoon Lee, Jaegul Choo, Dongmin Park, and Jongho Park. Can large language models develop strategic reasoning? post-training insights from learning chess. *arXiv preprint arXiv:2507.00726*, 2025a.

Dongyoon Hwang, Hojoon Lee, Jaegul Choo, Dongmin Park, and Jongho Park. Can large language models develop strategic reasoning? post-training insights from learning chess, 2025b. URL https://arxiv.org/abs/2507.00726.

Sai Dhanush Jannala. *Chess LLM Arena: A Framework for Evaluating Strategic Decision-Making in Large Language Models*. PhD thesis, Dublin, National College of Ireland, 2025.

Harsh Jhamtani, Varun Gangal, Eduard Hovy, Graham Neubig, and Taylor Berg-Kirkpatrick. Learning to generate move-by-move commentary for chess games from large-scale social forum data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1661–1671, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1154. URL https://aclanthology.org/P18-1154/.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

Kagi Search. llm-chess-puzzles: Benchmark llm reasoning capability by solving chess puzzles. GitHub repository, April 2025. URL https://github.com/kagisearch/llm-chess-puzzles. 1000 FEN puzzles; updated 2025-04-26. Accessed 2025-09-16.

Jaechang Kim, Jinmin Goh, Inseok Hwang, Jaewoong Cho, and Jungseul Ok. Bridging the gap between expert and language models: Concept-guided chess commentary generation and evaluation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 9497–9516, 2025.

Andrew Lee, Antonio Gulli, Bo Chang, Bob Fraser, Bovard Doerschuk-Tiberi, Chad Woodford, Chris Prichard, Chuck Sugnet, Daniel Hennes, Dima Yeroshenko, DJ Sterling, Elsa Dong, Hann Wang, Harrison Jobe, Ian Gemp, Jaimie Hwang, Jie Ren, John Schultz, Jon Lipovetz, Jun Peng, Justin Chiu, Karim Hakimzadeh, Kate Olszewska, Laurel Prince, Lloyd Hightower, Marc Lanctot, Martyna Plomecka, Meg Risdal, Meghan O'Connell, Minmin Chen, Nate Keating, Nenad Tomasev, Oran Kelly, Orhan Firat, Phoebe Kirk, Riley Jones, Roxanne Daniel, Ryan Trostle, Sahand Sharifzadeh, Siqi Liu, Timothy Chung, Tom Mason, Will Cukierski, Ya Xu, Yao Yan, Yi Su, Yuchen Zhuang, Yuting Han, and Yuexiang Zhai. Chess Text Input. https://www.kaggle.com/benchmarks/kaggle/chess-text, 2025. Google DeepMind, Google Cloud, Kaggle.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, et al. Websailor: Navigating super-human reasoning for web agent. *arXiv preprint arXiv:2507.02592*, 2025.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective, 2025. URL https://arxiv.org/abs/2503.20783.

Reid McIlroy-Young, Siddhartha Sen, Jon Kleinberg, and Ashton Anderson. Aligning superhuman ai with human behavior: Chess as a model system. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1677–1687, Virtual Event, CA, USA, 2020. ACM. doi: 10.1145/3394486.3403219. URL https://dl.acm.org/doi/10.1145/3394486.3403219.

Reid McIlroy-Young, Russell Wang, Siddhartha Sen, Jon Kleinberg, and Ashton Anderson. Learning models of individual behavior in chess. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1253–1263, 2022.

Daniel Monroe and Philip A Chalmers. Mastering chess with a transformer model. *arXiv preprint arXiv:2409.12272*, 2024.

Yu Nasu. Efficiently updatable neural-network-based evaluation functions for computer shogi. *The 28th World Computer Shogi Championship Appeal Document*, 2018. Original NNUE paper (in Japanese).

David Noever, Matt Ciolino, and Josh Kalin. The chess transformer: Mastering play using generative language models. *arXiv preprint arXiv:2008.04057*, 2020.

OpenAI. Openai o1 system card, 2024. URL https://arxiv.org/abs/2412.16720.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

Gian-Carlo Pascutto and Gary Linscott. Leela chess zero, 2018. URL http://lczero.org/.

Gian-Carlo Pascutto, Gary Linscott, et al. Leela zero: Go engine with no human provided knowledge. https://github.com/leela-zero/leela-zero, 2018.

Anian Ruoss, Grégoire Delétang, Sourabh Medapati, Jordi Grau-Moya, Li K Wenliang, Elliot Catt, John Reid, Cannada A Lewis, Joel Veness, and Tim Genewein. Amortized planning with large-scale transformers: A case study on chess. *Advances in Neural Information Processing Systems*, 37:65765–65790, 2024.

Maxim Saplin. Llm chess leaderboard. Project website, 2025. URL https://maxim-saplin.github.io/llm_chess/. Elo vs. calibrated engine and random baselines. Accessed 2025-09-16.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

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

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017a.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017b.

Stockfish Team. Stockfish: A strong open-source chess engine, 2023. URL https://stockfishchess.org/. NNUE integration since version 12 (2020).

Andreas Stöckl. Watching a language model learning chess. In Ruslan Mitkov and Galia Angelova (eds.), *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pp. 1369–1379, Held Online, September 2021. INCOMA Ltd. URL https://aclanthology.org/2021.ranlp-1.153/.

Richard Sutton. The bitter lesson. *Incomplete Ideas (blog)*, 13(1):38, 2019.

Colton Swingle, Henry Mellsop, and Alex Langshur. ChePT – applying deep neural transformer models to chess move prediction and self-commentary. CS224N course project report, Stanford University, 2021. URL https://web.stanford.edu/class/cs224n/reports/final_reports/report087.pdf. Stanford CS224N: Natural Language Processing with Deep Learning.

Zhenwei Tang, Difan Jiao, Blair Yang, and Ashton Anderson. Seam: Semantically equivalent across modalities benchmark for vision-language models. In *Second Conference on Language Modeling*.

Zhenwei Tang, Difan Jiao, Reid McIlroy-Young, Jon Kleinberg, Siddhartha Sen, and Ashton Anderson. Maia-2: A unified model for human-ai alignment in chess. *Advances in Neural Information Processing Systems*, 37:20919–20944, 2024.

Zhenwei Tang, Difan Jiao, Eric Xue, Reid McIlroy-Young, Jon Kleinberg, Siddhartha Sen, and Ashton Anderson. Learning to imitate with less: Efficient individual behavior modeling in chess. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=iw4kjcw319.

Shu Wang, Lei Ji, Renxi Wang, Wenxiao Zhao, Haokun Liu, Yifan Hou, and Ying Nian Wu. Explore the reasoning capability of llms in the chess testbed. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pp. 611–622, 2025.

Qianfeng Wen, Zhenwei Tang, and Ashton Anderson. Chessqa: Evaluating large language models for chess understanding. *arXiv preprint arXiv:2510.23948*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025a. URL https://arxiv.org/abs/2505.09388.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025b.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Hongyu Zang, Zhiwei Yu, and Xiaojun Wan. Automated chess commentator powered by neural chess engine. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5952–5961, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1597. URL https://aclanthology.org/P19-1597/.

Yiming Zhang, Athul Paul Jacob, Vivian Lai, Daniel Fried, and Daphne Ippolito. Human-aligned chess with a bit of search. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=bc2H72hGxB.

Yinqi Zhang, Xintian Han, Haolong Li, Kedi Chen, and Shaohui Lin. Complete chess games enable llm become a chess master. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pp. 1–7, 2025b.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization, 2025. URL https://arxiv.org/abs/2507.18071.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand, 2024. Association for Computational Linguistics. URL http://arxiv.org/abs/2403.13372.

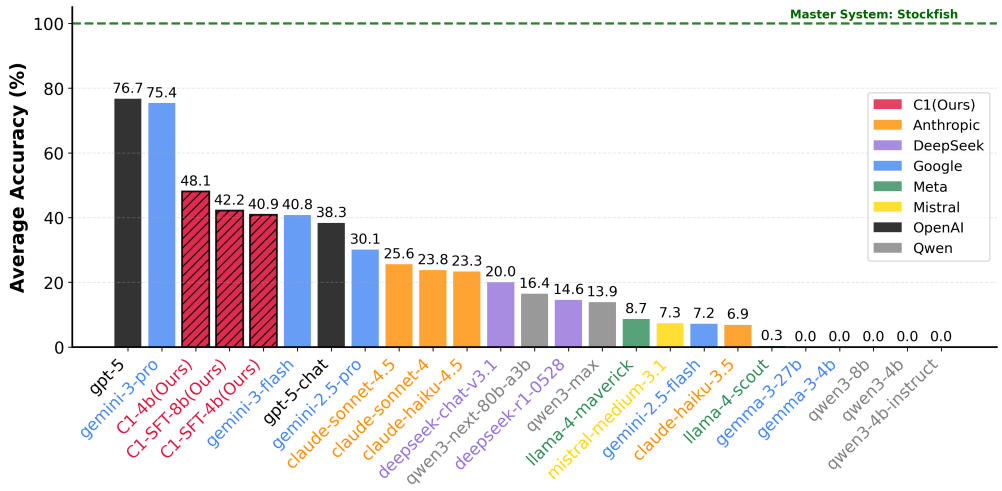


Figure 2: Chess puzzle-solving accuracy of C1 compared to frontier LLMs.

Master Distillation Prompt

General Instruction

```
You are a chess grandmaster generating training data for teaching small LLMs to solve chess puzzles.
CRITICAL: The small model only sees the FEN string. Your analysis must show how to go from FEN → piece positions → tactical relationships → solution. Ground everything in explicit square names.
```

Context (provided to teacher, hidden from student)

```
FEN: {fen}
Pieces: {piece_arrangement}
Side to Move: {side_to_move}
Opponent's Last Move: {last_move}
Solution: {move}
Rating: {rating}
Themes: {themes}
PVs: {principal_variation} (omitted for mateIn1)
Legal Moves: {legal_moves}
```

Task Description

```
TASK: Write natural chain-of-thought analysis arriving at {move}.
Your analysis should cover (in natural prose, vary the structure):
• Where key pieces are (use square names: ''the queen on h5'', ''king on g1'')
• What tactical relationship exists (attacks, pins, weak squares, defender counts)
• Why the move works (what it threatens, why opponent can't respond adequately)
Style:
• Objective voice, no ''I see/notice''
• Standard notation with brief clarification when helpful: ''Qxh7+ (queen takes h7 with check)''
• Never mention engine scores, ratings, themes, or that you were given the solution
• 4--10 sentences, scaled to complexity
End with exactly: FINAL_ANSWER: {move}
```

Table 5: RLVR Training Hyperparameters.

Hyperparameter	**Value**
Model Configuration	
Base Model	C1-SFT-4B
Algorithm	DAPO-C1
Gradient Checkpointing	Enabled
Data Configuration	
Max Prompt Length	1024
Max Response Length	512
Generation Batch Size	96
Training Batch Size	32
Rollout Samples	32
Sampling Parameters	
Temperature	1.0
Top-p	1.0
Top-k	-1
GPU Memory Utilization	0.8
Optimizer	
Learning Rate	1×10^{-6}
PPO Mini Batch Size	8
PPO Micro Batch Size (per GPU)	32
Log Prob Micro Batch Size	256
RL Specific	
KL Loss Coefficient	0.001
Entropy Coefficient	0
Clip Ratio (low / high)	0.2 / 0.28
Loss Aggregation	Token-mean
KL in Reward	Disabled
Filter Groups	Enabled (max batches: 5)
Training	
Total Epochs	1
Save Frequency	500
Test Frequency	10
Num GPUs	4
Early-Stop Tolerance	5

Table 6: Supervised Fine-Tuning Hyperparameters

Parameter	**Setting**
Base Model	Qwen3-4B-Instruct
Training Method	Full-parameter fine-tuning
Optimizer	AdamW (DeepSpeed ZeRO-3)
Learning Rate	1e$-$5
Batch Size	16×4 (grad accum) $\times$ 4 GPUs = 256
Epochs	10
Scheduler	Cosine with 10% warmup
Max Length	1024 tokens
Precision	BF16
Validation Interval	50 Steps
Early-Stop Tolerance	5

Table 7: Data statistics for SFT and RLVR training.

	SFT	**RLVR**
Total samples	39,609	39,585
Unique themes	66	66
Balanced themes (K)	50	50
Target per theme (M)	800	800
Rating range	399–3,154	399–3,196
Average rating	1,540	1,539

Table 8: Theme distribution in SFT and RLVR training data. We apply balanced sampling with $K = 50$ rare themes and target $M = 800$ samples per theme.

	endgame	mate	short	middlegame	crushing	long	advantage	mateIn2	oneMove	mateIn1	master	veryLong
SFT	20,817	17,971	17,205	16,210	12,957	10,236	7,858	6,930	6,856	6,824	6,133	5,294
RLVR	20,752	17,872	17,056	16,210	13,054	10,292	7,834	6,767	6,860	6,833	6,056	5,359

	sacrifice	advancedPawn	defensiveMove	kingsideAttack	fork	attraction	opening	pin	mateIn3	quietMove	rookEndgame	discoveredAttack	deflection	exposedKing	pawnEndgame	promotion
SFT	4,932	3,288	3,277	3,001	2,948	2,672	2,572	2,370	2,335	2,307	2,195	2,164	2,136	2,019	1,829	1,820
RLVR	5,037	3,316	3,178	2,957	2,939	2,718	2,614	2,364	2,395	2,278	2,159	2,245	2,279	2,005	1,850	1,901

	backRankMate	masterVsMaster	hangingPiece	queensideAttack	skewer	clearance	doubleCheck	mateIn4	intermezzo	queenEndgame	bishopEndgame	queenRookEndgame	zugzwang	attackingF2F7	trappedPiece	knightEndgame	mateIn5	capturingDefender
SFT	1,596	1,557	1,439	1,393	1,165	1,083	1,077	1,055	1,040	966	964	891	874	872	867	866	827	827
RLVR	1,594	1,587	1,515	1,425	1,122	1,081	1,061	1,047	1,053	967	973	898	892	871	874	850	830	819

	xRayAttack	cornerMate	interference	arabianMate	hookMate	anastasiaMate	equality	vukovicMate	blindSwineMate	triangleMate	superGM	enPassant	killBoxMate	castling	bodenMate	doubleBishopMate	dovetailMate	smotheredMate	balestraMate	underPromotion
SFT	825	818	817	810	808	806	805	805	805	805	803	803	802	800	800	800	800	800	697	512
RLVR	830	822	814	807	810	808	807	808	805	808	802	805	800	800	800	801	801	801	668	517

Table 9: Performance comparison across chess tactic themes. All values represent accuracy.

	advancedPawn	attraction	backRankMate	capturingDefender	defensiveMove	deflection	discoveredAttack	doubleCheck	fork	hangingPiece	mateIn1	mateIn2	pin	promotion	queensideAttack	sacrifice	skewer	trappedPiece	xRayAttack	zugzwang
Proprietary models																				
gpt-5	84	92	100	84	68	92	80	80	88	96	96	96	84	76	92	92	84	68	80	72
gemini-3-pro	80	68	92	80	68	68	84	72	88	80	84	84	76	68	96	88	84	56	80	68
gemini-3-flash	28	32	32	44	40	44	36	20	40	44	56	52	60	24	44	52	36	24	28	24
gpt-5-chat	28	40	68	36	44	48	32	36	48	32	44	64	32	40	44	52	48	28	40	32
gemini-2.5-pro	28	20	52	8	36	20	20	52	32	36	40	28	28	44	36	28	32	28	20	32
claude-sonnet-4.5	44	12	52	16	8	20	48	40	48	8	16	36	20	56	12	40	32	0	40	24
claude-sonnet-4	36	28	44	16	16	28	20	32	36	28	24	24	28	32	20	40	24	4	44	12
claude-haiku-4.5	40	28	48	12	16	40	32	32	48	20	32	20	20	36	24	24	12	0	20	8
gemini-2.5-flash	16	12	0	0	0	12	0	12	4	4	12	12	4	24	4	8	16	0	16	8
claude-haiku-3.5	8	8	8	4	8	0	4	16	8	16	8	12	8	8	0	4	8	0	8	4
Open-source models																				
deepseek-chat-v3.1	40	24	44	28	36	12	4	20	24	20	16	4	20	24	32	16	20	8	8	40
qwen3-next-80b-a3b	16	28	8	24	12	24	24	12	36	24	12	12	12	16	12	8	20	8	24	20
deepseek-r1-0528	16	20	4	24	24	24	12	16	32	36	12	4	12	20	8	8	16	8	4	20
qwen3-max	16	20	4	12	8	16	8	12	24	12	16	40	8	20	12	8	20	0	8	12
llama-4-maverick	24	0	4	8	8	4	4	16	12	12	12	8	16	4	4	0	12	0	12	12
mistral-medium-3.1	12	8	0	4	8	8	12	8	16	0	8	12	8	8	12	4	12	0	12	8
llama-4-scout	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
gemma-3-27b	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Ours																				
C1-SFT-4B	48	52	72	28	48	36	28	52	48	52	56	40	24	48	60	44	44	12	60	72
C1-SFT-8B	52	44	84	20	60	40	56	44	40	52	64	40	36	48	60	36	44	8	48	56
C1-4B	64	64	84	56	60	36	44	52	36	64	64	56	28	52	68	60	52	4	52	76

Algorithm 1 Theme-Balanced Data Sampling

Require: Dataset $\mathcal{D}$ where each puzzle p has theme set $\mathcal{T}(p)$
Require: Number of rare themes to balance K
Require: Maximum samples per theme M
Ensure: Balanced subset $\mathcal{D}_{\text{bal}}$
  Compute theme frequencies: $f(t) \leftarrow |\{p \in \mathcal{D} : t \in \mathcal{T}(p)\}|$ for all themes t
  Select rare themes: $\mathcal{T}_{\text{rare}} \leftarrow \arg\min_K f(t)$
  Initialize selected IDs: $\mathcal{S} \leftarrow \emptyset$
  Initialize output: $\mathcal{D}_{\text{bal}} \leftarrow \emptyset$
  for each theme $t \in \mathcal{T}_{\text{rare}}$ **do**
    $\mathcal{C}_t \leftarrow \{p \in \mathcal{D} : t \in \mathcal{T}(p) \wedge \text{id}(p) \notin \mathcal{S}\}$
    Sample $\min(M, |\mathcal{C}_t|)$ puzzles from $\mathcal{C}_t$ without replacement
    $\mathcal{D}_{\text{bal}} \leftarrow \mathcal{D}_{\text{bal}} \cup$ sampled puzzles
    $\mathcal{S} \leftarrow \mathcal{S} \cup \{\text{id}(p) : p \in$ sampled puzzles$\}$
  end for
  return $\mathcal{D}_{\text{bal}}$