Universiteit Leiden

Master Computer Science

Reinforcement Learning with Verifiable Rewards for Small Search Agents:

A Reward-Shape Ablation

Name:	Gaurisankar Jayadas
Student ID:	s4374886
Date:	10/06/2026
Specialisation:	Artificial Intelligence
1st supervisor:	Aske Plaat
2nd supervisor:	Álvaro Serra-Gómez

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

One way to make a language model better at reasoning is to let it practise: have it attempt a question, reward it only when its final answer is correct, and let Reinforcement Learning (RL) reshape its behaviour toward what works. Reinforcement Learning with Verifiable Rewards (RLVR) has driven much of the recent progress in Large Language Model (LLM) reasoning. DeepSeek-R1 (DeepSeek-AI et al., 2025) and OpenAI o1 (OpenAI, 2024) are well-known examples. The approach works best where an answer can be checked deterministically, such as in mathematics and coding. Search-R1 (Jin et al., 2025b), ReSearch (Chen et al., 2025), and R1-Searcher (Song et al., 2025) extend RLVR to *reason-over-search*: the model interleaves its own reasoning with calls to a search tool, and learns from the final answer alone. These results exist only at 3 billion to 32 billion parameters, mostly in the Qwen2.5 family, where even the sparse exact-match reward of Search-R1 suffices. It is not known whether the recipe survives below one billion parameters: a smaller model starts from a weaker prior, so it may fail to learn the behaviour at all, and the reward shape may matter more at this scale.

We test this directly. We train Qwen3.5-0.8B (Qwen Team, 2026) with GRPO (Shao et al., 2024) and an interleaved Wikipedia-search tool on a single multi-hop dataset (MuSiQue (Trivedi et al., 2022)), and evaluate every saved checkpoint on the held-out splits of a 7-benchmark question-answering suite. The reward is the only thing we vary: we compare three shapes (token-F1, a partial-credit overlap score; token-F1 with a small format floor; and exact-match-only), each over three random seeds.

Training is feasible. The best run reaches 0.352 average exact-match on the held-out suite, up from a 0.092 untrained floor, a gain of 3.8 times. At its best full-epoch checkpoint it is comparable to our reproduced previous-generation Search-R1 Qwen2.5-3B baseline (0.341) at a quarter of the parameters, a cross-generation comparison, not a controlled size study. With the rest of the recipe held fixed, the shape of the reward is a consequential design choice: the exact-match-only reward Search-R1 uses is worst on average at every seed at the matched training horizon (a ranking that would arise by chance with probability about 3.7%), and the dense F1 rewards drive the gains. We also see a reward-dependent efficiency effect: with no length penalty, the F1 policies issue fewer search calls and so use far fewer tokens overall (total tokens fall by a quarter to a half) even as held-out accuracy rises. Finally, a higher training reward does not translate into more exact matches: all three shapes plateau at a training-time exact-match rate of about 20% to 22%.

The reason-over-search recipe transfers below one billion parameters with no distillation step from a larger teacher during training, in contrast to compact agentic-search results that recover the behaviour only with distillation guidance (Kotoge et al., 2026). With the rest of the recipe held fixed, the reward shape is the decisive lever: it separates the best of the three shapes from the worst, though all three clear the untrained floor by a wide margin. Many GRPO modifications exist to improve learning, yet plain, unmodified GRPO with the right reward is enough here. For practitioners this is a concrete, low-cost recipe for a small reason-over-search agent, with a warning: the sparse exact-match re-

ward, RLVR's default in mathematics and coding, is the wrong starting point at small scale. The main limitation is scope: the effect, though consistent across three seeds, rests on a single model; a second model size and a same-generation control remain future work.

Acknowledgements

This thesis was carried out in collaboration with Alstom Netherlands B.V., where I undertook the internship that ran alongside the project. I am grateful to the company for the opportunity, and for the practical context and resources that shaped this research.

My special thanks go to Alireza Khanshan and Thijs van der Burg, whose support throughout the internship was steady and generous, and made this work both possible and enjoyable.

I am also grateful to my two thesis supervisors, Aske Plaat and Álvaro Serra-Gómez, for their guidance, careful feedback, and patience over the course of this project.

I gratefully thank CloudRift for the compute grant that made this work's training and held-out evaluation possible.

Contents

List of Figures 4

List of Tables 6

1 Introduction 8
- 1.1 Topic . . . 8
- 1.2 Problem Statement and Research Questions . . . 9
- 1.3 Method . . . 9
- 1.4 Contributions . . . 10
- 1.5 Overview . . . 12

2 Related Work 13
- 2.1 Reinforcement Learning for LLM Reasoning . . . 13
- 2.2 Retrieval-Augmented Reasoning with Tool Use . . . 14
- 2.3 Reward Shaping . . . 15
- 2.4 Efficiency in Agentic RL (Emergent vs Imposed Compression) . . . 16
- 2.5 Small-model Post-training . . . 17

3 Method 18
- 3.1 Task and Data . . . 18
- 3.2 Model and Tool Interface . . . 19
- 3.3 Training: GRPO on NeMo-RL . . . 20
- 3.4 Reward Shapes Ablated . . . 21
- 3.5 Retriever . . . 24
- 3.6 Training Horizon and Seeds . . . 24
  - 3.6.1 Seed Reporting Protocol . . . 24
- 3.7 Evaluation Protocol . . . 24
- 3.8 Pilot Study: Design Decisions Carried Into the Main Study . . . 25

4 Results 26
- 4.1 Held-out QA performance across reward shapes and seeds . . . 26
  - 4.1.1 Per-seed trajectories . . . 26
  - 4.1.2 Cross-seed summary (matched horizon) . . . 26
  - 4.1.3 Best-checkpoint numbers . . . 27
  - 4.1.4 What this section establishes (interpretation) . . . 29
- 4.2 Cross-generation comparison . . . 33
- 4.3 Training dynamics and emergent compression (mechanism) . . . 34
  - 4.3.1 Per-seed trajectories . . . 35
  - 4.3.2 Cross-seed view (seed-robustness of the dynamics) . . . 36
  - 4.3.3 Cross-seed average (matched horizon) . . . 37
  - 4.3.4 Emergent compression (key insight) . . . 37
  - 4.3.5 From cold start to lean to planning: three rollouts . . . 40

4.3.6 Higher reward is not more exact matches: the 20 to 22 % training-time EM ceiling . . . 41
4.3.7 Oscillations (explore-and-lean) . . . 42
4.3.8 What the joint read establishes (and what is missing) . . . 42

5 Limitations and Future Work **43**
5.1 Limitations . . . 43
5.2 Future Work . . . 44

6 Conclusions **45**
6.1 Feasibility verdict . . . 45
6.2 The optimised-recipe answer . . . 45
6.3 Cross-generation: post-training versus parameter count . . . 46
6.4 Broader implications . . . 47

Bibliography **49**

A Full hyperparameter tables (training + eval) **54**
A.1 Training hyperparameters (GRPO recipe, all nine runs) . . . 54
A.2 Evaluation hyperparameters (held-out suite) . . . 55
A.3 Retriever and index choice (IVF-SQ8 vs flat) . . . 57
A.4 The training-software overlay (component inventory) . . . 58
A.5 GRPO vs PPO optimiser memory (why no value critic) . . . 59

B Prompt templates and tool-call format **60**

C Reward design and construction **62**

D Held-out eval detail (backs Results 4.1) **64**
D.1 Per-benchmark EM at each run's best checkpoint . . . 64
D.2 Exact per-leg end-steps and the matched-horizon definition . . . 65
D.3 avg F1 and ACC per run . . . 65
D.4 Per-dataset best-checkpoint spread . . . 66
D.5 Full per-checkpoint sweeps . . . 66
D.6 F1 and ACC trajectory figures + consolidated three-metric table . . . 66
D.7 Statistical read of the three-seed ranking . . . 73
D.8 Held-out search-count profile and the termination gap . . . 73

E Cross-generation and paper-reproduction detail (backs Results 4.2) **76**
E.1 Paper targets vs our reproduction (Qwen2.5-3B Search-R1 GRPO) . . . 76
E.2 Cross-generation: untrained floor + best trained, both generations . . . 77

F Training-health diagnostics (backs Results 4.3) **78**
F.1 Cadence-to-cadence stability (all nine runs) . . . 80
F.2 Per-cadence reward-shape triad at seed 44 (C1-C31) . . . 82

G Qualitative rollout traces **84**
G.1 One broad search, right answer (reward 1.0) vs its failing sibling (reward 0.0) 84
G.2 Explicit numbered planning, clean 3-hop chain (reward 1.0) . . . 85
G.3 Cold-start failure (reward 0.0) . . . 86

H Pilot study (Qwen3-0.6B) full results **87**
H.1 Prompt sensitivity and the 0.1-floor finding 87
H.2 Trained vs untrained eval (two 0.6B checkpoints) 87
H.3 Base model cannot bootstrap 88
H.4 In-distribution tags are free at matched step count 88

I Prompt-mode ablation (selecting the terse prompt) **90**

J Chain-flip audit (LLM judge) **91**

K Public artifacts **94**

L Compute and cost **95**

List of Figures

3.1 The training pipeline: the critic-free GRPO loop (a) and one reason-search-read rollout (b). 22

4.1 Seed 42 held-out EM by reward shape (to step 180); F1+format leads and all three reward shapes clear the 0.092 floor early. 27
4.2 Seed 43 held-out EM by reward shape (to step 230). 28
4.3 Seed 44 held-out EM by reward shape (full epoch, to step 310); F1-only reaches the run-high 0.352 at step 310 and EM-only's late rise is length-driven. 29
4.4 Cross-seed mean with min-to-max range per reward shape, matched horizon. 30
4.5 Matched-horizon average EM, grouped by reward shape, one bar per seed. 31
4.6 Matched-horizon cross-seed mean of EM, F1, and ACC by reward shape. . 32
4.7 Seed 42 training dynamics, capped at the standard horizon step 180 (per-seed caps are 180 / 230 / 312 for seeds 42 / 43 / 44, with the last full cadence at step 310). 35
4.8 Seed 43 training dynamics, capped at the standard horizon step 230. . . . 36
4.9 Seed 44 training dynamics (full epoch). 37
4.10 F1-only training dynamics across seeds 42, 43, and 44. 38
4.11 F1+format training dynamics across seeds 42, 43, and 44. 39
4.12 EM-only across seeds 42, 43, and 44; the late over-search re-entry is visible only at the full seed-44 horizon. 40
4.13 Cross-seed mean training dynamics per reward shape, matched horizon. . . 41

D.1 Seed 42 token-F1 (to step 180). 67
D.2 Seed 43 token-F1 (to step 230). 68
D.3 Seed 44 token-F1 (full epoch, to step 310). 68
D.4 Cross-seed mean ± range, token-F1, matched to step $\leq$ 180. 69
D.5 Matched-horizon best token-F1, grouped by reward shape, one bar per seed. 69
D.6 Seed 42 ACC (to step 180). 70
D.7 Seed 43 ACC (to step 230). 70
D.8 Seed 44 ACC (full epoch, to step 310). 71
D.9 Cross-seed mean ± range, ACC, matched to step $\leq$ 180. 71
D.10 Matched-horizon best ACC, grouped by reward shape, one bar per seed. . 72
D.11 Held-out behaviour by search count at each run's best-EM checkpoint. . . 74

F.1 Seed 42: gradient norm, loss, KL penalty, approx. entropy, aborted ratio, natural-termination rate. 79
F.2 Seed 43, same six panels. 79
F.3 Seed 44 (full epoch), same six panels. 80
F.4 The 2x A100 TP=2 run. Healthy to ~step 200, then entropy collapses and the gradient norm explodes to ~131 by step 289; reward dies to 0 by step 313. The DP=4 run of the same recipe and seed did not collapse. 81

H.1 Three runs identical except the user prompt (X/Y/Z). Tool use settles at 2 / 1 / 0 calls per episode (centre) while reward converges to ~0.16 to 0.19 (left), every run above the 0.1 format floor (dashed). 87
H.2 The two base-model runs that trained the full 2301 steps: tool calls stay at 0 (left) and reward stays flat near 0 (centre). The two other base runs stopped early and are omitted for legibility. 88
H.3 The two best-of-block pilot runs. Out-of-distribution `<search>`/`<result>` (grey) and in-distribution `<tool_call>`/`<tool_response>` JSON (blue) converge to nearly the same reward (~0.17 to 0.19), both above the 0.1 floor (dashed); the out-of-distribution run only reaches its quoted 0.215 ceiling by training far longer. 89

List of Tables

3.1 The terse user prompt. 19

4.1 Matched horizon (best checkpoint at step ≤ 180), average EM across the three seeds per reward shape. 27
4.2 Best checkpoint at each seed horizon (seed 42 at step ≤ 180, seed 43 at step ≤ 230, seed 44 at step ≤ 312), average EM per reward shape. 28
4.3 Matched horizon, average F1 per reward shape at each run's best-F1 checkpoint, showing the same ordering on the dense metric. 28
4.4 Substring-cover accuracy (ACC) at each run's seed-horizon best-EM checkpoint, per reward shape; EM-only is worst on the lenient metric as well. . . 29
4.5 Floor-versus-trained average EM on the same eval network for the older-generation Qwen2.5-3B and the current-generation Qwen3.5-0.8B, with the Reinforcement Learning (RL) lift for each. 34
4.6 Training-side run-high reward versus run-high exact-match rate (%R=1) at seed 44 for the F1-only, F1+format, and EM-only runs. 41

A.1 GRPO training recipe (per-knob value and rationale) shared by all nine runs of the Qwen3.5-0.8B search-tool ablation. 54
A.2 Held-out evaluation hyperparameters: decode, retrieval, and context settings applied identically to every checkpoint of every run. 56
A.3 Retriever index comparison: resident RAM, recall@10, score fidelity, and per-query CPU latency for the exact flat index versus the default IVF4096-SQ8 index. 57
A.4 Optimiser-memory accounting for the 0.8B policy: PPO with a value critic versus GRPO with a group-relative baseline and no critic. 59

C.1 How our F1+format reward differs from ReSearch's: the same reward-shape components mapped onto our tool-call surface syntax. 63

D.1 Per-benchmark EM at each run's seed-horizon best checkpoint, with average F1 and ACC at that checkpoint, for all nine reward-shape by seed runs. 64
D.2 Exact final-checkpoint step per leg (reward shape by seed); the basis for the matched-horizon definition. 65
D.3 Average token-F1 at each run's best-F1 checkpoint (matched horizon / seed horizon), per reward shape and seed. 65
D.4 Average substring-cover ACC at each run's seed-horizon best-EM checkpoint, per reward shape and seed. 65
D.5 Per-dataset best-checkpoint spread: for each run, the best-average checkpoint and how many of the seven datasets peak there. 66
D.6 Consolidated EM, F1, and ACC at each run's matched-horizon (step ≤ 180) best-average-EM checkpoint. 72

D.7 Consolidated EM, F1, and ACC at each run's seed-horizon best-average-EM checkpoint. 72
D.8 Completed-rollout exact-match by search count. 75

E.1 Per-benchmark EM: Search-R1 paper targets versus our Qwen2.5-3B reproduction, for the base and instruct GRPO checkpoints. 76
E.2 Cross-generation untrained floor, best trained checkpoint, and RL lift on the same eval network, for Qwen2.5-3B and Qwen3.5-0.8B. 77

F.1 Cadence-to-cadence stability of held-out average EM across the nine runs (mean and max absolute step-to-step change, std, and worst single drop), sorted most-stable first. 81
F.2 Per-cadence training reward and exact-match rate (%R=1) for the three reward shapes at seed 44, isolating the reward axis. 82

1 Introduction

1.1 Topic

Large Language Models (LLMs) acquire broad competence from next-token pretraining (Radford et al., 2019), but their most striking recent gains in multi-step reasoning come from Reinforcement Learning (RL) applied after pretraining. OpenAI's o1 (OpenAI, 2024) showed, in a closed model, that large-scale RL over long chains of thought produces reasoning that keeps improving with additional test-time computation; DeepSeek-R1 (DeepSeek-AI et al., 2025) demonstrated the same in an open model and showed that the behaviour can be incentivised by RL alone. What makes this training scale is that the reward is verifiable: for domains with checkable answers, such as mathematics and code, a deterministic checker scores the output directly, so no learned reward model is needed, unlike Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022), which trains a reward model from human preference comparisons. This paradigm, Reinforcement Learning with Verifiable Rewards (RLVR) (Lambert et al., 2025), is at the core of most of the recent progress in LLM reasoning.

A reasoning model is still bounded by what its weights encoded at training time. It cannot answer questions about facts it never saw or that have since changed, and when pressed it tends to fabricate plausible but wrong answers. Retrieval-Augmented Generation (RAG) (Lewis et al., 2020) addresses this by letting the model condition its output on documents fetched from an external corpus, but a single retrieve-then-read step is weak for questions that require several linked facts, where what to look up next depends on what has already been worked out.

The natural combination is to let the model interleave its own reasoning with search: it decides when and what to query, reads the retrieved evidence, reasons further, and only then answers, with the whole behaviour trained by RLVR against an answer-match reward. Search-R1 (Jin et al., 2025b), ReSearch (Chen et al., 2025), and R1-Searcher (Song et al., 2025) established that this works, training a policy to call a search tool mid-reasoning over a frozen Wikipedia corpus and learning the reason-over-search behaviour from an outcome reward alone. These results, however, are reported almost exclusively at 3B to 32B parameters, on previous model generations.

Smaller models are far cheaper to train and to serve, and arguably the more suitable class for agentic systems (Wang et al., 2024; Belcak et al., 2025), but they carry a much weaker prior, and the little evidence at sub-1B scale is cautionary: a study of compact search agents (Kotoge et al., 2026) finds that pure RL is too unstable to bootstrap the behaviour from a cold start at 0.5B to 1B and recovers it only with distillation from a larger teacher. A weaker prior also means a model leans more heavily on how informative its reward is, so the shape of that reward (sparse versus dense) plausibly becomes a more decisive design choice at small scale than at the scales studied so far. Whether the reason-over-search recipe (the full training configuration: policy model, GRPO hyperparameters, retriever, prompt, and reward) transfers to a sub-1B model trained from its own initialisation, and which reward shape works best at that scale, is open. This thesis

takes up exactly that question.

1.2 Problem Statement and Research Questions

Problem statement. RL with verifiable rewards over a search tool (Search-R1 (Jin et al., 2025b), ReSearch (Chen et al., 2025)) is established for 3B-to-32B models; whether it transfers to a sub-1B model is open. A smaller model has a much weaker prior and may fail to learn the interleaved reason-over-search behaviour, and the reward shape (the functional form that turns an answer's correctness into a scalar reward, ranging from sparse to dense) may be more decisive at this scale. This thesis asks whether a sub-1B model can be trained for reason-over-search question answering (QA), and which reward shape works best.

Main RQ. Can RL with verifiable rewards over a search tool be made to work on a sub-1B language model, and what reward shape is best?

Three primary research questions decompose it:

- **RQ1 (feasibility and generalisation):** can Group Relative Policy Optimization (GRPO) (Shao et al., 2024) with a search tool train a sub-1B model (Qwen3.5-0.8B (Qwen Team, 2026)) to beat its untrained baseline on held-out multi-hop QA?
- **RQ2 (reward shape, accuracy):** which reward shape (sparse Exact Match (EM): a 0/1 reward that fires only on an exactly correct answer; dense token-F1: partial credit for word overlap with the correct answer; or denser F1 with a format floor) gives the best held-out exact-match, and does the partial-credit floor help or mask the tool-use signal at sub-1B scale?
- **RQ3 (reward shape, stability):** which reward shape is the most stable across random seeds and training horizon?

Three further questions are reported as observations rather than objectives we set out to optimise; each is reported in the Results chapter and revisited in the Conclusions:

- **RQ4 (training dynamics, emergent compression):** how do tool-use and response length evolve under the unmodified outcome reward, and what drives any change when the reward carries no explicit length penalty?
- **RQ5 (post-training versus parameter count):** does RLVR on a newer-generation sub-1B model match or exceed a roughly four times larger previous-generation model on held-out multi-hop QA; that is, does the generation-plus-RLVR advantage outweigh the parameter-count advantage?
- **RQ6 (reward versus exact-match):** does a higher training reward correspond to a higher exact-match rate, or can a reward shape inflate the optimisation scalar without producing more exact matches?

Scope. The questions are phrased as "sub-1B" with Qwen3.5-0.8B as the single instance; single-model scope is a limitation (Chapter 5), not a claim about all sub-1B models.

1.3 Method

We train Qwen3.5-0.8B (Qwen Team, 2026) (the hybrid variant, since the base model cannot bootstrap the tool-call format from a cold start, a 0.6B pilot finding in Appendix H.3)

with GRPO (Shao et al., 2024) on MuSiQue (Trivedi et al., 2022), the single hardest multi-hop dataset, interleaving `<think>`, `<tool_call>` search, and `<tool_response>` in one reasoning chain, and we evaluate held-out on a 7-benchmark QA suite (NQ (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), PopQA (Mallen et al., 2023), HotpotQA (Yang et al., 2018), 2Wiki (Ho et al., 2020), MuSiQue, and Bamboogle (Press et al., 2023)). The core experiment is a controlled ablation that crosses three reward shapes with three random seeds, holding everything else fixed so that the reward shape is the only varied axis. Because the model, training data, prompt, retriever, and optimiser settings are identical across all nine runs, differences in held-out performance (measured as EM, F1, and accuracy, defined in Chapter 3) and in training stability can be attributed to the reward shape itself rather than to some other difference between runs. The full recipe, including the policy model, the GRPO hyperparameters, the Qwen3.5-native tool-call format, the prompt, and the floor-drop rationale, is given in Chapter 3.

1.4 Contributions

In plain terms, the work makes three main contributions, with a set of further contributions. First, Reinforcement Learning (RL) with a search tool can teach a sub-1B model to interleave its own reasoning with live retrieval and answer multi-hop questions: every reward shape we test clears the untrained baseline by a wide margin, and it does so with no distillation from a larger teacher during training. Second, and most striking, the exact-match reward (the form RLVR inherits from the mathematics and coding settings it grew out of) turns out to be the wrong choice at this scale: with the rest of the recipe held fixed, the sparse exact-match reward is the worst of the three shapes at every seed at the matched training horizon, and the dense partial-credit (token-F1) reward is what carries the learning. Third, we explain why. A dense F1 reward grades every attempt, so the policy always has an informative signal to improve on, whereas the sparse exact-match reward leaves most attempts scoring alike, which collapses the group-relative advantage GRPO learns from; the training metrics bear this out, with the exact-match reward producing the weakest gradient updates of the three shapes at the matched training horizon (Chapter 4 and the training-health appendix). Around these sit further contributions: the smaller model's best held-out accuracy is comparable to a previous-generation model roughly four times larger (a cross-generation comparison, not a controlled study of model size); the better-rewarded policies grow leaner over training, searching less and using fewer tokens overall even though the reward carries no length penalty; a higher training reward does not by itself mean more exact answers; and all three reward shapes plateau at the same accuracy ceiling, set by the scale of the model and data rather than by the reward.

These contributions, each tied to one or more of the research questions stated above, are listed exactly below: the three main contributions first, then the further contributions.

Main contributions.

- **C1 (RQ1): the reason-over-search recipe transfers below one billion parameters.** GRPO with a search tool trains Qwen3.5-0.8B (Qwen Team, 2026), from its own initialisation and with no distillation from a larger teacher, to clear its untrained floor by a wide margin on the held-out 7-benchmark suite. This contrasts with the only prior sub-1B evidence, where pure RL is too unstable to bootstrap the behaviour and the result is recovered only with distillation (Kotoge et al., 2026).

- **C2 (RQ2, RQ3): at this scale the exact-match reward is the wrong choice.** We run a controlled reward-shape ablation (F1-only, F1+format, and EM-only; three seeds each; the full held-out 7-benchmark evaluation). The central result is a negative one: the Search-R1-faithful EM-only reward, the exact-match form that is standard in the mathematics and coding settings RLVR grew out of, is the worst shape at every seed, measured at the matched training horizon; under a no-effect null this ranking would arise by chance with probability about 0.037. The dense token-F1 reward is what drives the gains. Between the two F1 shapes the ordering depends on the seed: F1+format is the most robust across seeds, while F1-only reaches the highest single-run score.
- **C3 (RQ2, RQ3): why the exact-match reward underperforms.** Under GRPO the update is driven by the within-group (group-relative) advantage, so when a prompt's rollouts nearly all score alike that signal collapses. The sparse 0/1 exact-match reward leaves most groups in exactly that regime, whereas the dense token-F1 reward grades every rollout and keeps an informative spread. The training metrics confirm it: at the matched horizon the exact-match reward produces the weakest gradient updates of the three shapes and is the only shape whose policy entropy stays pinned high rather than sharpening (Chapter 4 and the training-health appendix).

Further contributions.

- **C4 (RQ1): a reproducible training and evaluation pipeline.** A policy is trained with GRPO and a search tool, and every checkpoint along the way is evaluated on the held-out benchmarks. Because the existing verl framework (Sheng et al., 2024) does not support Qwen3.5, we port the training loop to NeMo-RL (NVIDIA NeMo Team, 2025). We then validate the evaluation side of the pipeline by reproducing the published Search-R1 (Jin et al., 2025b) Qwen2.5-3B GRPO numbers to within 2.5 percentage points (Appendix E.1).
- **C5 (RQ2): the partial-credit format floor aids discovery.** The 0.1 format floor aids discovery rather than masking the tool-use signal at 0.8B scale, which reverses what our earlier 0.6B pilot had predicted. An LLM-judge audit on the seed-42 run then locates where the floor's advantage comes from: compared with the floorless F1-only policy, the F1+format policy produces more answers in a parseable format and more answers grounded in the retrieved passages, though its reasoning chains are no more coherent (Chapter 4).
- **C6 (RQ4): an emergent-compression effect, with its mechanism.** Under an unmodified outcome reward, the F1 shapes become leaner tool-callers over training: they search less often and their total token count falls by roughly a quarter to a half, even as held-out EM rises. The likely cause is the fixed generation cap acting as an implicit length penalty that the dense F1 reward can exploit and the sparse EM-only reward cannot. The shrinking search count and total-token budget run opposite to the growing budget reported for Search-R1 and ReSearch at larger scale (the per-rollout reasoning length itself follows the same early-dip-then-rise shape Search-R1 reports).
- **C7 (RQ5): a cross-generation comparison.** A current-generation sub-1B model plus RLVR matches or slightly exceeds a roughly four times larger previous-generation 3B model on held-out multi-hop QA. We treat this as a capability-frontier

observation, not a controlled comparison: the smaller model is also a newer generation (different pretraining data, post-training, and architecture) and was trained with a different RL recipe, so the result cannot be attributed to parameter count, model generation, or recipe alone.

- **C8 (RQ6): a decoupling between training reward and exact-match.** A higher training reward does not imply more exact matches: the 0.1 floor in F1+format inflates the reward scalar without lifting answer accuracy. In addition, all three reward shapes plateau at the same training-time exact-match rate of roughly 20% to 22%, a ceiling that appears to be a property of the model-and-data scale rather than of the reward shape.

1.5 Overview

The remainder of this thesis is organised as follows. Chapter 2 (Related Work) surveys RL for LLM reasoning, retrieval-augmented reasoning with tool use, reward shaping, efficiency in agentic RL, and small-model post-training. Chapter 3 (Method) details the policy model and the GRPO recipe, the MuSiQue training setup, the Qwen3.5-native tool-call format, the held-out 7-benchmark evaluation protocol, the three reward shapes, and the prompt together with the floor-drop rationale. Chapter 4 (Results) is organised by research question: Section 4.1 reports held-out QA across reward shapes and seeds (RQ1 to RQ3), Section 4.2 reports the cross-generation comparison (RQ5), and Section 4.3 reports the training dynamics, emergent compression, and the reward-versus-exact-match ceiling (RQ4 and RQ6). Chapter 5 (Limitations and Future Work) discusses the single-model and single-dataset scope, the training-horizon limits, the seed count, and the EM and F1-only metrics, and it proposes a same-generation control and a 2B scale test as future work. Chapter 6 (Conclusions) answers each research question and states the feasibility verdict together with the recipe recommendation: which reward shape, and which recipe choices, carry the result at this scale.

2 Related Work

This chapter situates the thesis in five lines of work. Section 2.1 reviews the Reinforcement Learning (RL) recipe that trains language models to reason. Section 2.2 turns to the work that connects that reasoning to an external search tool. Section 2.3 covers the debate over how much reward engineering such training needs. Section 2.4 examines how reasoning effort and tool use evolve during training. Section 2.5 addresses the specific challenges of post-training very small models. Each section closes by stating how the prior work differs from the sub-1B reason-over-search setting studied here.

2.1 Reinforcement Learning for LLM Reasoning

Reinforcement Learning entered Large Language Model (LLM) post-training as a way to align models with human intent. InstructGPT (Ouyang et al., 2022) established the now-standard pipeline of supervised fine-tuning, a learned reward model over human preference rankings, and policy optimisation with Proximal Policy Optimization (PPO) (Schulman et al., 2017); its headline result, that a 1.3B aligned model is preferred to the 175B base model, made Reinforcement Learning (RL) post-training a decisive rather than cosmetic step. PPO itself, a first-order clipped-surrogate method, remains the optimiser this whole lineage builds on.

The recent shift is from rewards learned from human preferences to rewards that are verifiable: a deterministic checker scores the answer, removing the reward model entirely. OpenAI's o1 (OpenAI, 2024) demonstrated, in a closed model, that large-scale Reinforcement Learning (RL) over long chains-of-thought (CoT) yields reasoning that improves with test-time compute. DeepSeek-R1 (DeepSeek-AI et al., 2025) showed the open counterpart, and its R1-Zero variant established the strong claim that reasoning can be incentivised by Reinforcement Learning (RL) alone, with no supervised reasoning traces, using simple rule-based verifiable rewards. The reward paradigm itself was crystallised as Reinforcement Learning with Verifiable Rewards (RLVR) in open recipes such as Tulu 3 (Lambert et al., 2025), where a verification function replaces the reward model for any domain with checkable answers. The reasoning these methods elicit is the explicit step-by-step form introduced by Chain-of-Thought (CoT) prompting (Wei et al., 2022), now produced by the policy rather than demonstrated in the prompt.

The algorithm at the centre of this thesis, Group Relative Policy Optimization (GRPO), was introduced with DeepSeekMath (Shao et al., 2024). GRPO drops PPO's value critic and instead estimates each response's advantage from a group-relative baseline (the mean and standard deviation of rewards over a sampled group of responses to the same prompt), which removes the value critic (typically a model as large as the policy) and so roughly halves the memory and compute of training while keeping PPO's clipped objective. This thesis differs from the work above in the character of its verifiable reward and in scale. The headline RLVR results, DeepSeek-R1 (DeepSeek-AI et al., 2025) and DeepSeekMath (Shao et al., 2024), are reported in mathematics and code, where the verifiable reward is a near-unambiguous automatic check (symbolic answer-matching in

math, unit-test execution in code), and almost entirely at 1.5B parameters and above: DeepSeek-R1's smallest released policy is its 1.5B distillation, and the small-model RLVR literature surveyed in Section 2.5 floors at the same size. Our setting moves RLVR to open-domain question answering, where the verifiable signal is a noisy answer-match against a reference, and down to a sub-1B policy whose weaker prior may not support the same training dynamics. The retrieval-augmented line of work that supplies the task itself is surveyed separately in Section 2.2.

2.2 Retrieval-Augmented Reasoning with Tool Use

Grounding reasoning in external knowledge has a long pre-Reinforcement Learning history. Retrieval-Augmented Generation (RAG) (Lewis et al., 2020) coupled a parametric generator with a non-parametric corpus retrieved by dense passage retrieval, improving factuality on knowledge-intensive tasks but using a single retrieve-then-read step. Multi-hop questions, where what to retrieve next depends on what has already been derived, exposed the limits of one-shot retrieval and motivated a family of prompted methods that interleave retrieval with reasoning. ReAct (Yao et al., 2023) interleaves free-form reasoning with tool actions so that observations ground the next thought; Self-Ask (Press et al., 2023) decomposes a question into explicit, separately answered follow-ups (and contributes the Bamboogle benchmark used in our evaluation); IRCoT (Trivedi et al., 2023) alternates retrieval and Chain-of-Thought sentence by sentence; FLARE (Jiang et al., 2023) makes retrieval active by deciding when to retrieve from a prediction of the upcoming sentence; and Self-RAG (Asai et al., 2024) learns adaptive retrieval and self-critique through reflection tokens. Toolformer (Schick et al., 2023) showed, in parallel, that a model can learn when and how to call tools in a self-supervised way. All of these either hand-engineer the control flow in the prompt or learn it through supervision rather than from an outcome reward.

The Reinforcement Learning (RL) turn replaced that hand-engineering with a single outcome reward. Search-R1 (Jin et al., 2025b) trains a policy with PPO or GRPO to interleave reasoning with multi-turn search-engine calls over a frozen corpus, masking retrieved tokens from the policy-gradient loss and rewarding only exact-match on the final answer; it is the reference recipe and the baseline this thesis reproduces. ReSearch (Chen et al., 2025) trains the same interleaved behaviour with GRPO and a token-F1 reward carrying a small partial-credit format floor, and shows that training on a single multi-hop dataset generalises across benchmarks; it is the recipe this thesis ports. R1-Searcher (Song et al., 2025) reaches comparable results with a two-stage outcome-based Reinforcement Learning (RL) scheme and, notably, with no supervised cold start; it also reports a small answer-reward ablation, comparing exact-match, cover-exact-match, and token-F1, and finds token-F1 best at 7B. A 2025-to-2026 wave of work argues that the outcome reward is too sparse and adds dense, per-step credit: HiPRAG (Wu et al., 2026) rewards the necessity of each search decision and sharply cuts over-searching, while a cluster of methods (IGPO (Wang et al., 2026), TIPS (Xie et al., 2026), IG-Search (Liang et al., 2026), Search-P1 (Xia et al., 2026)) adds per-turn or per-search information-gain or path-coverage rewards. IG-Search reports the largest same-task contrast with the outcome-only recipe (average EM 0.430 against the Search-R1 baseline's 0.312 on the same seven benchmarks), although only part of that gap is attributable to the step-level reward itself, the rest to the surrounding recipe.

Two gaps separate this work from ours. First, every Reinforcement Learning (RL)

trained reason-over-search result above sits between 3B and 32B parameters, almost all on previous-generation model families: the Qwen2.5 series in Search-R1 and ReSearch, with R1-Searcher also training a Llama-3.1 model of the same vintage. Whether the recipe transfers to a sub-1B model with a much weaker prior is untested; by a weaker prior we mean less factual knowledge and weaker instruction-following stored in the weights, although a newer generation's pretraining data, post-training, and architecture partially offset the deficit. Second, none of these recipes makes the reward shape itself the controlled variable. Search-R1 fixes a sparse exact-match reward (and explicitly leaves richer reward terms to future work); ReSearch fixes a dense token-F1 reward with a format floor; and R1-Searcher's exact-match-versus-F1 comparison noted above is a single-seed side ablation at 7B on three of the paper's four benchmarks, not a controlled study at sub-1B. This thesis instead crosses three reward shapes (sparse exact-match, dense token-F1, and token-F1 with a format floor) with three seeds, holds everything else fixed, and evaluates held-out on all seven benchmarks, so that the reward shape is the axis under test. The wider field's momentum, meanwhile, is toward adding dense per-step credit, which makes an outcome-only reward a minority position; this thesis deliberately stays at the outcome level and treats the dense-shaping cluster as the foil against which a minimal-shaping recipe is positioned rather than a component it adopts. In one line, the niche is the reward-shape question for outcome-level reason-over-search, answered at the sub-1B frontier and from the model's own post-trained initialisation, with no distillation step in training.

2.3 Reward Shaping

GRPO has spawned a family of stabilising modifications. DAPO (Yu et al., 2025a) is the most validated open trick stack: decoupled clip bounds to preserve exploration, dynamic sampling that drops prompts whose group is entirely right or entirely wrong, a token-level policy-gradient loss, and a soft length penalty, reproducing frontier reasoning at 32B with everything open-sourced. Dr. GRPO (Liu et al., 2025) instead diagnoses two biases hidden in the objective, a response-length normalisation that inflates length (especially for wrong answers) and a question-difficulty normalisation that over-weights too-easy and too-hard prompts, and removes both for an unbiased, more token-efficient estimator. GSPO (Zheng et al., 2025) moves the importance ratio and clipping from the token level to the sequence level, with its documented gains concentrated in mixture-of-experts training.

A counter-current questions whether any of this is needed at small scale. JustRL (He et al., 2025) shows that a single-stage, essentially plain GRPO recipe with one fixed hyperparameter set, reused unchanged across two 1.5B models, matches or beats elaborate multi-stage pipelines at a fraction of their compute, and that standard additions can actively hurt: an explicit length penalty collapses exploration, and even a more lenient verifier degrades the result by removing useful formatting pressure. Closest to this thesis, a recent study of how to train deep-research agents (Xu et al., 2026) runs a direct reward ablation in the Search-R1 setting and finds that reward shape is decisive: a naive token-F1 reward underperforms exact-match through an answer-avoidance collapse, and only an added action penalty lifts F1 above EM.

This debate is conducted almost entirely in mathematics and code at 1.5B and above (JustRL (He et al., 2025) at 1.5B; DAPO (Yu et al., 2025a) at 32B; GSPO (Zheng et al., 2025) on a roughly 30B mixture-of-experts), and where it touches search it does so at 3B and larger. The reward-shape question for retrieval-augmented question answering at

sub-1B scale is open. JustRL's "a minimal recipe can win" is the design rationale for this thesis: we vary only the reward shape and keep plain GRPO otherwise, so any effect is attributable to the reward rather than to a stack of interacting tricks. Our result agrees with this literature that reward shape is the decisive lever, and it supplies the missing sub-1B data point: at 0.8B the dense token-F1 reward beats exact-match. This matches the 7B answer-reward ablation in R1-Searcher (Song et al., 2025), where token-F1 also beats exact-match, and contrasts with the naive-token-F1-below-exact-match result reported at 3B and above (Xu et al., 2026), where an unprotected token-F1 reward triggers an answer-avoidance collapse that only an added action penalty undoes. Token-F1 thus beats exact-match in two of the three reward ablations across scales (7B in R1-Searcher and our 0.8B) and loses only where the naive-token-F1 answer-avoidance collapse fires (3B and above, in the Search-R1 setting). Our own floorless F1-only reward is itself unprotected, yet it does not trigger that collapse at 0.8B and still beats exact-match. The contradiction therefore tracks neither model scale (F1 wins at both 0.8B and 7B but loses at 3B) nor the presence of a format floor (our floorless F1 does not collapse), but whether the answer-avoidance failure mode is triggered at all.

2.4 Efficiency in Agentic RL (Emergent vs Imposed Compression)

Over the course of training, a policy changes how much it reasons and how often it calls its search tool. The literature treats these dynamics in two ways. One line imposes brevity directly: RoRecomp (Li et al., 2025) compresses reasoning by recomposing rollouts rather than penalising length, optimal-tool-call methods (Wang et al., 2025c) reward calling fewer tools, and HiPRAG penalises unnecessary search. A second line studies length as an emergent property. Concise Reasoning (Fatemi et al., 2025) shows analytically that the default effect of Reinforcement Learning (RL) is to grow response length (incorrect answers push the policy toward longer outputs), so that explicit length reduction requires a dedicated secondary phase rather than emerging on its own. An empirical study of reasoning-over-search agents (Jin et al., 2025a) finds that search-call frequency during training is shaped by retriever quality, non-monotonically: a useless retriever teaches the agent to abandon search, a weak sparse retriever drives escalating call counts to compensate, and a strong dense retriever yields a moderate, judicious number of calls. A separate analysis (Yu et al., 2025b) treats the length and call dynamics as observational under a mildly call-rewarding objective.

This thesis reports an effect that sits against both lines. Under an unmodified outcome reward, with no length penalty and no brevity incentive, the dense-reward policies become leaner tool-callers and shorten their rollouts (total tokens fall by roughly a quarter to a half) as held-out accuracy rises, while the sparse exact-match policy fails to compress. The driver is the fixed generation cap acting as an implicit length penalty, a mechanism that the dense reward exploits and the sparse reward does not. Where prior work either imposes brevity directly (through a length penalty, a tool-call budget, or rollout recomposition) or observes length growth as the Reinforcement Learning (RL) default, the compression here is emergent and reward-shape-conditioned, inverting the length-growth narrative of the reason-over-search recipes we build on.

2.5 Small-model Post-training

Most RLVR results sit at 1.5B parameters and above; DeepSeek-R1's distillations (DeepSeek-AI et al., 2025), JustRL (He et al., 2025), ExGRPO (Zhan et al., 2026), Tina (Wang et al., 2025a), and 1-Shot RLVR (Wang et al., 2025b) all floor at 1.5B. The sub-1B regime has received far less attention and behaves differently. Open post-training recipes such as Tulu 3 establish the SFT-then-DPO-then-RLVR template at 8B to 405B, and the model family used here, Qwen3.5 (Qwen Team, 2026), provides an open hybrid policy that can reason deliberately or answer directly. The evidence specific to small agentic-search models is sparse and cautionary. A study of compact search agents (Kotoge et al., 2026) finds that pure Reinforcement Learning (RL) is too unstable to bootstrap agentic retrieval from a cold start at 0.5B to 1B, where poor initial capability gives sparse rewards and training collapses early, and recovers the behaviour only with teacher cold-start initialisation and continuous distillation guidance, reaching an average exact-match in the low thirties at 0.5B against a 3B teacher. Our own pilot study is consistent with this regime: a 0.6B base model (Qwen Team, 2025) could not bootstrap the tool-call format from cold start, which is why the main study uses a post-trained hybrid initialisation.

This thesis differs from the small-model agentic-search literature in its lever. Where the compact-search-agent study above stabilises sub-1B training with distillation from a larger teacher, this study shows that a current-generation sub-1B model can be trained using GRPO for reason-over-search from its own post-trained initialisation, without distillation, and that the reward shape is the decisive choice. It also situates the outcome against a previous-generation model roughly four times larger, a cross-generation comparison developed in Chapter 4 and framed there, with its confounds, as a capability-frontier observation rather than a controlled scaling law.

3 Method

We open with the conventions that govern every subsection below; Figure 3.1 overviews the full training pipeline (the critic-free GRPO loop and one reason-search-read rollout). We train a Group Relative Policy Optimization (GRPO) baseline that normalises rewards by their group mean and standard deviation (Shao et al., 2024). Each run realises at most one epoch, up to about 312 optimiser steps (the shorter legs stop earlier; see Section 3.6). Our reward parity with ReSearch (Chen et al., 2025) is value-equivalent rather than byte-identical: we drop the `\boxed{}` answer gate, as detailed in the reward-shape section and Appendix C. Finally, the chat template auto-injects a tool-schema system block; the byte-exact rendered prompt is reproduced in Appendix B, and the full hyperparameter tables are in Appendix A.

3.1 Task and Data

Our task is open-domain multi-hop question answering with an interleaved retrieval tool. The policy emits search queries, reads retrieved Wikipedia chunks, and produces a final answer, alternating reasoning and retrieval until it commits to an answer span.

We train on MuSiQue (Trivedi et al., 2022) only. MuSiQue is a 2-to-4-hop dataset and the hardest benchmark in the reference papers' evaluations (trained models score lowest on it; Search-R1 evaluates on seven benchmarks, ReSearch on four multi-hop ones), and we use it as a single-dataset training corpus by design so that generalisation is measured strictly on held-out data. The held-out evaluation suite comprises seven benchmarks: the single-hop Natural Questions (NQ) (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), and PopQA (Mallen et al., 2023); the 2-hop HotpotQA (Yang et al., 2018) and Bamboogle (Press et al., 2023); and the 2-to-4-hop 2WikiMultiHopQA (2Wiki) (Ho et al., 2020) and MuSiQue (Trivedi et al., 2022), totalling 51,713 rows per checkpoint.

We state four explicit hypotheses, each mapping to a research question. They make the claims that the results sections must support concrete and falsifiable.

- **H1**: a sub-1B-parameter model can be trained using GRPO above the untrained floor on held-out QA (RQ1).
- **H2**: reward shape changes held-out exact-match (EM) at fixed model, data, and retriever (RQ2).
- **H3**: the reward-shape effect is seed-stable (RQ3).
- **H4**: the training-reward scalar and the exact-match rate can diverge, so a reward shape can raise reward without raising EM (RQ6).

RQ4 (emergent compression) and RQ5 (post-training versus parameter count) are observational questions (Section 1.2) and carry no pre-stated hypothesis; they are reported descriptively in Sections 4.3 and 4.2 respectively.

Use the `search` tool to look up facts as needed. When you have the answer, write it inside `<answer>` and `</answer>`. For example, `<answer> Beijing </answer>`. Question: `{question}`

Table 3.1: The terse user prompt: the entire hand-authored part of the template, wrapping the question. The auto-injected system block and the byte-exact full render are in Appendix B.

3.2 Model and Tool Interface

Our policy is Qwen3.5-0.8B (Qwen Team, 2026), a post-trained hybrid model (it can either reason step by step before answering or answer directly). We deliberately avoid calling it "distilled", since no primary source confirms it: the dense Qwen3.5 models are documented by a blog post and model cards rather than a technical report, and none of those states the training recipe of the small variants. The previous generation is explicit on this point: the Qwen3 technical report (Qwen Team, 2025) describes its small models, including the Qwen3-0.6B our pilot study trains, as produced by strong-to-weak distillation from larger teachers. We do not assume the same holds for Qwen3.5.

The tool interface is the first place our design departs from the reference recipes. We use the Qwen3.5-native nested-XML `<tool_call>` and `<tool_response>` tags, which are in-distribution for the model, rather than the invented string tags of the reference recipes, which are off-distribution for Qwen3.5: both papers have the policy emit its query between `<search>` tags, with the retrieved passages returned between `<information>` tags in Search-R1 (Jin et al., 2025b) and `<result>` tags in ReSearch (Chen et al., 2025). The pilot study showed that in-distribution tags cost nothing at equal step count when compared at a matched horizon (Appendix H.4); this finding justifies the tag choice adopted here. We also ensure byte-alignment: the training overlay imports the reward scorer, the tool-call parser, and the prompt template from the evaluation package rather than re-implementing them, so training rollouts and held-out evaluation render byte-identical prompts, tags, and retrieval formatting from one shared code object. Trained checkpoints are therefore directly evaluable; without this guarantee, any score difference could hide a formatting drift between two implementations of the same prompt, and ruling that out would require auditing the two pipelines against each other.

The training template is given in full as a byte-exact rendered prompt in Appendix B; we summarise it here. The user message is a terse instruction wrapping the question, reproduced in Table 3.1.

The system role carries only the auto-injected `# Tools` JSON-schema block; the search tool is wired as a Qwen3.5 function schema rather than described in prose. The policy emits searches in Qwen3.5-native nested-XML (a `<tool_call>` containing an inner `<function=search>`) and reads retrieved chunks as `<tool_response>` turns. This differs from both Search-R1 and ReSearch, which describe the tool in plain prose and use the invented `<search>` query tags (returning passages between `<information>` tags in Search-R1 and `<result>` tags in ReSearch); ReSearch additionally wraps the final answer in `\boxed{}`.

3.3 Training: GRPO on NeMo-RL

We take the Reinforcement Learning with Verifiable Rewards (RLVR) to GRPO background from the related-work discussion in Section 2.1, which traces the lineage from InstructGPT (Ouyang et al., 2022) through Proximal Policy Optimization (PPO) (Schulman et al., 2017) to the reasoning models o1 (OpenAI, 2024) and R1 (DeepSeek-AI et al., 2025), and finally to GRPO as introduced with DeepSeekMath (Shao et al., 2024), whose group-relative baseline drops PPO's value critic. Here we give only the GRPO specifics of our run.

On the framework, the pilot study ran on verl (Sheng et al., 2024) with Qwen3-0.6B (Qwen Team, 2025). The main study moved to NeMo-RL (NVIDIA NeMo Team, 2025) with Qwen3.5-0.8B because verl does not support Qwen3.5; this is a port forced by tooling, not a preference. Training runs on NeMo-RL, and the retriever is a separate HTTP service described in the retriever section below.

For the objective, a group of G rollouts is sampled per prompt, and the policy maximises a clipped surrogate with an additive Kullback-Leibler (KL) penalty to a frozen reference:

$$J(\theta) = \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\min\Big(\rho_i A_i,\ \mathrm{clip}(\rho_i,\,1-\varepsilon,\,1+\varepsilon)\,A_i\Big) - \beta\, D_{\mathrm{KL}}(\pi_\theta \,\|\, \pi_{\mathrm{ref}})\right],$$

with the importance ratio and the group-relative advantage

$$\rho_i = \frac{\pi_\theta(y_i \mid x)}{\pi_{\mathrm{old}}(y_i \mid x)}, \qquad A_i = \frac{r_i - \mathrm{mean}(\{r_j\})}{\mathrm{std}(\{r_j\})}.$$

We use $G = 5$, $\varepsilon = 0.2$, and $\beta = 1\mathrm{e}{-3}$ with the k3 KL estimator, and the group-mean plus standard-deviation baseline; the full recipe is in Appendix A.1. This is the GRPO of DeepSeekMath.

We choose GRPO partly for its memory profile. The group-relative baseline removes PPO's value critic, which would otherwise be a second policy-sized network of roughly 0.8B parameters with its own weights, gradients, and Adam optimiser moments; avoiding it matters on modest hardware. The optimiser-state arithmetic is worked through in Appendix A.5.

The rollout follows a string-driven reason-search-read loop. Generation runs until a stop string (`</tool_call>` or `</answer>`) or on end-of-sequence; on a tool call, the retriever service is queried (top-5, as described in the retriever section), each of the five retrieved chunks is injected as its own `<tool_response>` block, the five blocks stacked inside a single user turn, and generation resumes. The rollout ends on `<answer>`, end-of-sequence, or a non-binding cap of 10 turns and 8192 tokens, which corresponds to about 5 to 7 turns in practice. This is the same loop as ReSearch (end-of-sequence-terminated, top-5) and Search-R1 (budget-bounded, top-3), run here with a colocated vLLM generation engine.

We mask retrieved tokens from the loss. The GRPO loss and its KL term are computed only over model-generated tokens (the reasoning, tool-call, and answer spans); retrieved `<tool_response>` passages are masked out. In our pipeline this is achieved by message-role typing: the mask is assigned per message from its role, so every token of a "tool" turn receives a loss-mask of 0 and every token of an assistant turn receives 1. Both reference papers mask retrieved tokens: Search-R1 ablates the choice and reports a substantially

worse average EM when training on retrieved tokens, and ReSearch adopts the same masking without an isolated ablation.

Our NeMo-RL overlay is our first software contribution (C4). Stock NeMo-RL provides the GRPO trainer, the rollout loop, and generic environment interfaces, with bundled example tasks for mathematics, code execution, and games; it ships no retrieval-augmented QA task, and neither does verl (Search-R1's release bundles its own pieces as a verl-based repository pinned to the previous model generation). Our overlay supplies that task layer: a dataset adapter for the Search-R1-format MuSiQue parquet (the reference recipes' published data schema, into which our MuSiQue training corpus is converted); the multi-turn search-tool environment, an HTTP client to the FlashRAG (Jin et al., 2025c) and E5 (Wang et al., 2022) retriever service that parses each tool call, fetches passages, and appends them as tool turns until the rollout answers or hits its caps; the reward modules for the three shapes (described in the reward-shape section); and the Qwen3.5 tool-call parsers and tool schema. The reward, parser, and tool schema are re-exported from the evaluation pipeline, so training and evaluation share one code object; this is the byte-alignment guarantee discussed in the model and evaluation sections. The complete component inventory, including the registry and processor glue and the three small framework patches, is in Appendix A.4.

To summarise the setup, with full tables in Appendices A and E: the model is Qwen3.5-0.8B; training uses MuSiQue only; evaluation uses the seven benchmarks of the task-and-data section; baselines are the untrained floor and our Search-R1 Qwen2.5-3B GRPO reproduction (within plus or minus 2.5 percentage points of the paper, Appendix E.1); the metrics are EM, F1, and accuracy (ACC); training is on NeMo-RL (data-parallel degree 4 with a colocated vLLM engine, Appendix L); and retrieval uses FlashRAG with E5-base-v2 over the Dense Passage Retrieval (DPR) (Karpukhin et al., 2020) Wikipedia-2018 corpus with a top-5 IVF-SQ8 index, as described next. The training retriever stack is identical to ReSearch's (E5-base-v2 with the DPR Wikipedia-2018 corpus and FlashRAG, top-5); the Search-R1 reproduction uses the same corpus and encoder at the paper's top-3 (its pipeline keeps the FlashRAG top-3 default, while ours overrides to top-5), so the calibration baseline shares its retrieval stack with the system it calibrates up to retrieval depth, the one disclosed depth difference (Appendix A.2).

3.4 Reward Shapes Ablated

We ablate three single-axis reward variants, all computed on the last `<answer>` block after SQuAD normalisation (Rajpurkar et al., 2016); full pseudocode and the ReSearch-versus-ours difference table are in Appendix C.

All three build on two standard ways of scoring an answer against the gold answer, both computed on normalised text. Exact match (EM) is all-or-nothing: it is 1 only if the answer matches a gold answer word-for-word, and 0 otherwise. Token-F1 instead gives partial credit for word (token) overlap. Counting the words the predicted answer and a gold answer share, precision is the share of the predicted words found in the gold answer and recall is the share of the gold words found in the prediction; F1 combines the two as their harmonic mean, $\mathrm{F1} = 2\,(\mathrm{precision} \times \mathrm{recall})/(\mathrm{precision} + \mathrm{recall})$, which is high only when both are high. We take the best F1 over the gold-answer list (the exact, repeat-aware definitions are in Section 3.7). The three reward shapes turn these two measures into a training signal:

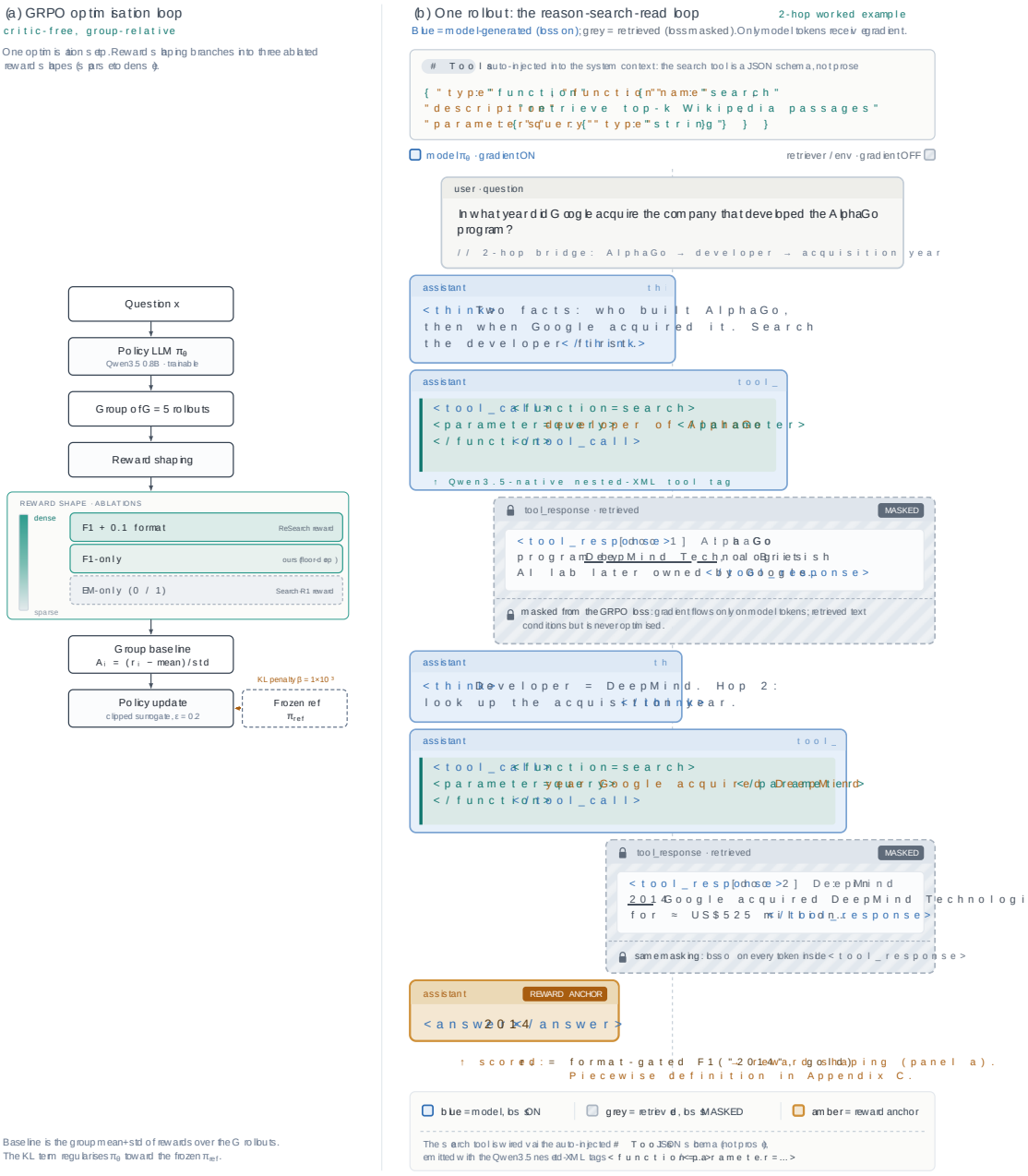


Figure 3.1: The training pipeline. (a) The critic-free GRPO loop: a group of `G=5` rollouts per question is scored, the reward node branches into the three ablated shapes (dense F1 plus 0.1 format, F1-only, and sparse EM-only); the group mean plus standard deviation sets the advantage baseline `A_i=(r_i-mean)/std` with no value critic (Appendix A.5), and the policy update is a clipped surrogate (`epsilon=0.2`) regularised by a KL penalty (`beta=1e-3`) to the frozen reference. (b) One reason-search-read rollout on an illustrative two-hop example: the model interleaves `<think>` reasoning and Qwen3.5-native `<tool_call>` searches (wired by the auto-injected `# Tools` JSON schema) with retrieved `<tool_response>` passages, ending in `<answer>`; the retrieved spans are shaded to mark that the GRPO loss is computed only on model-generated tokens, and the final answer is scored by the reward shape and fed back into the loop. The transcript is schematic; real MuSiQue traces are in Appendix G.

- **F1-only**: the reward is SQuAD token-level F1, or 0 if no parseable answer is present; there is no floor.
- **F1 plus 0.1 format**: the reward is 0 if the format is invalid; 0.1 if the format is valid but F1 equals 0; otherwise F1. The 0.1 is a floor that replaces F1, not an additive bonus (an F1 of 0.5 yields a reward of 0.5). This shape is reward-identical to ReSearch but drops the `\boxed{}` gate, making it value-equivalent rather than byte-identical.
- **EM-only (0/1)**: the reward is 1 on an exact match and 0 otherwise. This is the paper-faithful form from Search-R1.

Writing $\hat{a}$ for the answer parsed from the last `<answer>` block (absent if there is none), $\mathrm{fmt}(o)$ for the format-validity check on a rollout o, and EM, F1 for the metrics of Section 3.7 (with Y the gold-answer set), the three reward shapes are

$$r_{\mathrm{EM}}(o) = \mathrm{EM}(\hat{a}, Y), \qquad r_{\mathrm{F1}}(o) = \mathrm{F1}(\hat{a}, Y), \tag{3.1}$$

both 0 when $\hat{a}$ is absent, and the floored shape, identical to ReSearch's reward (Chen et al., 2025),

$$r_{\mathrm{F1+fmt}}(o) = \begin{cases} \mathrm{F1}(\hat{a}, Y), & \mathrm{F1}(\hat{a}, Y) > 0 \\ 0.1, & \mathrm{F1}(\hat{a}, Y) = 0 \ \wedge \ \mathrm{fmt}(o) \\ 0, & \mathrm{F1}(\hat{a}, Y) = 0 \ \wedge \ \neg\,\mathrm{fmt}(o). \end{cases} \tag{3.2}$$

There are two provenance anchors and one departure. The F1 plus 0.1 format shape is exactly ReSearch's reward (F1 if F1 is greater than 0, else 0.1 if the format is valid, else 0). The EM-only shape is Search-R1's outcome reward (exact-match, with no format term). The F1-only shape is our own floor-drop ablation and departs from both. ReSearch additionally requires the answer to be wrapped in `\boxed{}`; we drop that gate, which makes our F1 plus 0.1 format shape value-equivalent to ReSearch (Appendix C).

The organising principle of the ablation is a reward-density axis. The three shapes run from sparse to dense. EM-only is a sparse 0/1 signal that grants credit only on an exact match. F1-only is a dense, graded token-overlap signal. F1 plus format keeps that dense F1 signal and adds a format-validity floor of 0.1 for a well-formed but wrong answer. The experiment asks where on this density axis a sub-1B policy learns best. A meaning-level (semantic-similarity) reward would extend the axis further and is left to future work (Section 5.2).

The motivation for the floor-drop comes from the 0.6B pilot, and we keep it in the main text because it sets up a key result. A pilot finding observed that the partial-credit 0.1 floor used by ReSearch masks the tool-use signal at 0.6B; this is precisely why the F1-only arm drops the floor. The pilot prompt-sensitivity result that surfaced this and the related prompt-versus-reward findings is in Appendix H.1. We keep this in the body, rather than buried in the appendix, because it sets up the headline results inversion: the floor turns out to help at 0.8B.

We keep the prompt-versus-reward attribution clean. The floor is a reward finding, whereas the prompt was fixed by its own two-step ablation before the reward ablation began, so that all three reward runs share one prompt (Appendix I). In that ablation, dropping a verbose system-protocol default for a minimal user message lifted the untrained model's EM 6.6 times, and a 12-variant selection then chose the terse form, worth a further roughly 1.7 times over minimal. The 6.6 times lift therefore belongs to the minimal-versus-verbose step, not to the terse prompt itself; Appendix I carries the full data and attribution.

3.5 Retriever

We use the Wikipedia-2018 corpus with the E5-base-v2 encoder and an IVF4096-SQ8 quantised index (Douze et al., 2024) for both training and evaluation. The reference papers retrieve with an exact flat inner-product index; we use it nowhere, for two budget reasons. Under training rollout load the flat index times out, and serving it needs roughly 65 GB of host RAM or a GPU-resident copy, spare capacity our rented boxes did not have. The quantised index fits in about 16 GB at a measured recall@10 of 96 to 99% against the flat index; Appendix A.3 details the index choice and its retrieval-fidelity implications.

3.6 Training Horizon and Seeds

The training horizon is roughly one epoch, about 312 steps per run on MuSiQue. Runs differ in realised length, as detailed in the seed-reporting protocol below. We use three seeds per reward shape (42, 43, and 44). Compute provenance, including hardware and wall-clock, is in Appendix L; we keep it out of the main body by design.

3.6.1 Seed Reporting Protocol

Runs differ in length: the seed-42 leg reaches 180 steps, the seed-43 legs reach 230 steps, and the seed-44 legs reach 312 steps (the exact per-leg end-steps are in Appendix D.2). We therefore adopt the following reporting conventions.

Cross-seed claims use the matched horizon (the best checkpoint at step ≤ 180, the largest horizon common to all nine legs) rather than best-of-run, because best-of-run is length-biased; we still report best-of-run, but we flag it explicitly when we do. The body carries one 3x3 (reward-shape by seed) summary table at a matched horizon, with a per-reward-shape mean plus or minus range across the three seeds as the seed-robustness measure. Per-seed, per-checkpoint tables and the three-seed trajectory overlays are deferred to the appendix. Finally, the ranking of the two F1 shapes flips with the seed (F1+format wins seeds 42 and 43, F1-only wins seed 44); since averaging across seeds would erase that disagreement, we report the per-seed ordering explicitly alongside the cross-seed mean.

3.7 Evaluation Protocol

We use greedy decoding, matching the evaluation convention of both reference recipes' official code: Search-R1's (Jin et al., 2025b) benchmark validation path disables sampling, and ReSearch's (Chen et al., 2025) released evaluation configuration does the same (neither paper states the evaluation decoding in its text). The primary metrics are EM and F1, with substring-cover accuracy (ACC) reported in the appendix. All three are computed on the extracted answer span after SQuAD-style normalisation (Rajpurkar et al., 2016) (lower-casing and stripping articles, punctuation, and extra whitespace), and all are evaluation metrics, defined independently of the training rewards that happen to reuse two of them. EM is 1 if the normalised prediction exactly equals a normalised gold answer, else 0. F1 is the token-level harmonic mean of precision and recall between the normalised prediction and gold tokens. ACC is 1 if a normalised gold answer appears as a substring of the normalised prediction, else 0 (more lenient than EM).

Concretely, write $\mathcal{N}(\cdot)$ for the normalisation above, let $\hat{a}$ be the extracted answer span, $Y = \{y_1, \ldots, y_k\}$ the set of gold answers, and $n_t(s)$ the number of times token t occurs in $\mathcal{N}(s)$. With per-gold token overlap $c_y = \sum_t \min\{n_t(\hat{a}), n_t(y)\}$, precision $P_y = c_y / \sum_t n_t(\hat{a})$, and recall $R_y = c_y / \sum_t n_t(y)$,

$$\mathrm{EM}(\hat{a}, Y) = \begin{cases} 1, & \exists\, y \in Y : \ \mathcal{N}(\hat{a}) = \mathcal{N}(y) \\ 0, & \text{otherwise,} \end{cases} \qquad \mathrm{F1}(\hat{a}, Y) = \max_{y \in Y} f_y, \quad f_y = \begin{cases} \dfrac{2\, P_y R_y}{P_y + R_y}, & c_y > 0 \\ 0, & c_y = 0. \end{cases} \tag{3.3}$$

Each checkpoint is evaluated over 51,713 rows using the byte-aligned prompt shared with training.

3.8 Pilot Study: Design Decisions Carried Into the Main Study

This short bridge subsection records the pilot's influence on the main study. The pilot (Qwen3-0.6B (Qwen Team, 2025) on verl (Sheng et al., 2024)) fixed four design decisions used in the main study; the full pilot results are in Appendix H.

First, the hybrid model converges slowly but stably (reward clusters at 0.18 to 0.22), so a hybrid (instruct) base is viable for GRPO with search (Finding 1). Second, the base model cannot bootstrap the tool-call format cold (4 of 4 attempts stayed at 0 tool calls), so the main study uses the hybrid rather than the base (Finding 2; full detail in Appendix H.3). Third, prompt drives behaviour more than reward, and the 0.1 floor masks the tool-use signal, which together motivate the prompt design and the F1-only floor-drop arm (Findings 3 and 4; the two figures referenced in the model and reward-shape sections). Fourth, in-distribution tags cost nothing at equal step count (Finding 5), which justifies the tag scheme.

The full pilot results, the trained-versus-untrained 0.6B evaluation, and the base-cannot-bootstrap finding are in Appendix H: the trained-versus-untrained evaluation in Appendix H.2 and the bootstrap failure in Appendix H.3.

4 Results

We present results eval-first and mechanism-after: the held-out performance across reward shapes and seeds comes first, the cross-generation comparison follows, and the training dynamics and emergent-compression analysis close the chapter. This chapter keeps the trajectory shape and the interpretable emergent-compression finding; the deep training-health diagnostics (gradient norm, KL divergence, entropy, clip ratio, loss, and the tensor-parallel collapse) are deferred to Appendix F.

4.1 Held-out QA performance across reward shapes and seeds

We evaluate nine Group Relative Policy Optimization (GRPO) (Shao et al., 2024) runs (three reward shapes times three seeds) on Qwen3.5-0.8B (Qwen Team, 2026), evaluating each checkpoint on the 51,713-row held-out 7-benchmark suite with greedy decoding and byte-aligned prompts. Every exact-match (EM) number reported here is the 7-benchmark average. The dashed line in every figure is the untrained floor, an average EM of 0.092. All runs train on MuSiQue (Trivedi et al., 2022) only, so six of the seven benchmarks are strictly out-of-distribution and MuSiQue is the single in-distribution dataset entering the average (flagged as "train" in the per-benchmark table, Appendix D.1); the held-out generalisation claim therefore rests on the six out-of-distribution benchmarks.

The standard horizons differ by seed: seed 42 runs to 180 steps, seed 43 to 230, and seed 44 to 312 (with checkpoints saved to step 310). The per-seed panels show each seed to its own horizon. Because the seeds differ in length, we compare cross-seed numbers at the matched horizon (best checkpoint at step ≤ 180), the common minimum across all three. Exact per-leg end-steps and the full untruncated sweeps are in the held-out Appendix D.

4.1.1 Per-seed trajectories

Figures 4.1 to 4.3 show one seed each: each panel overlays the three reward shapes, plotting held-out average EM against checkpoint step, against the floor. The same per-seed and cross-seed figures for token-F1 and substring-cover accuracy (ACC), together with a consolidated EM/F1/ACC table at each run's best checkpoint, are in Appendix D.6; the reward-shape ordering there matches the EM figures here (the metric-robustness check discussed in Results 4.1.4).

4.1.2 Cross-seed summary (matched horizon)

Two figures summarise the three seeds at the matched horizon (best checkpoint at step ≤ 180). Figure 4.4 plots each reward shape's cross-seed mean with its minimum-to-maximum band, so the band width reads directly as seed-robustness; Figure 4.5 shows

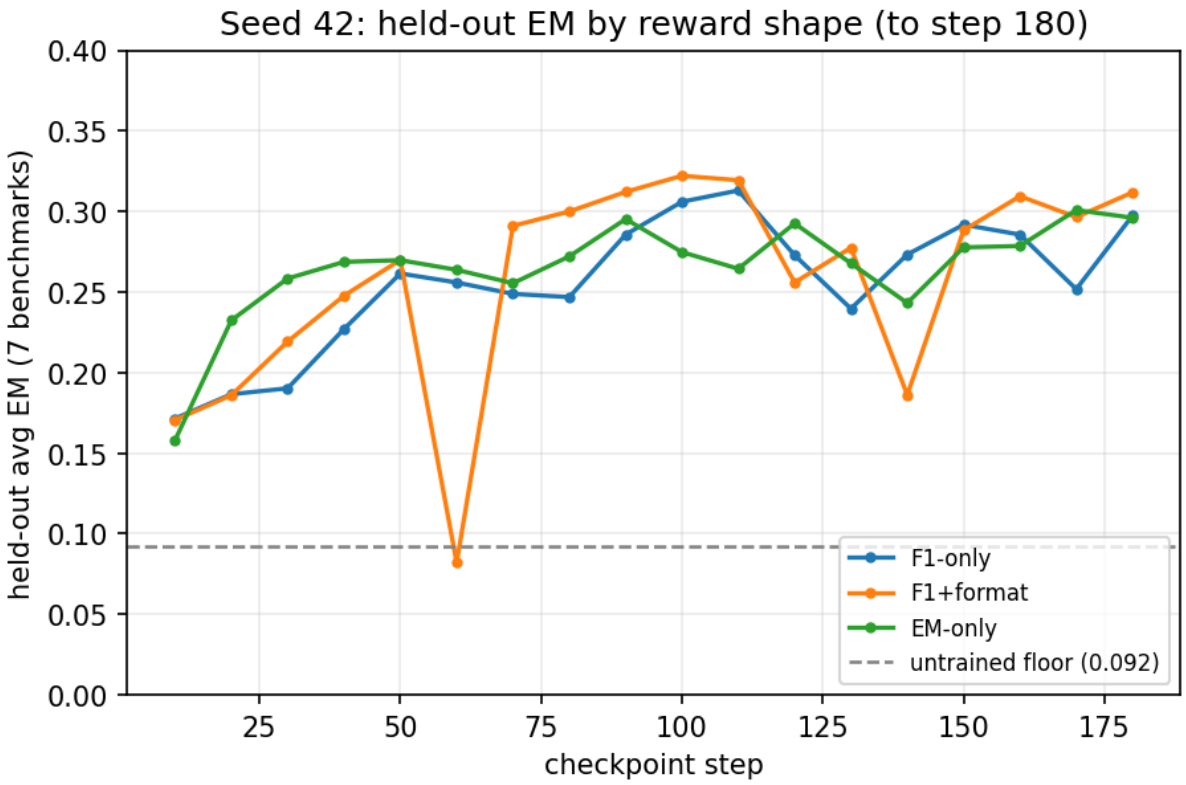


Figure 4.1: Seed 42 held-out EM by reward shape (to step 180); F1+format leads and all three reward shapes clear the 0.092 floor early.

the same data as one bar per seed. F1+format has the tightest band (most seed-robust), EM-only's mean sits lowest, and F1-only's band is widened by the seed-43 collapse.

4.1.3 Best-checkpoint numbers

We compute the following from the per-checkpoint evaluation tables. We report two views. The matched horizon is the best checkpoint at step ≤ 180, the honest cross-seed anchor common to all seeds. The seed horizon is the best checkpoint within each seed's own length (seed 42 at step ≤ 180, seed 43 at step ≤ 230, seed 44 at step ≤ 312). Per-benchmark breakdowns are in Appendix D.1, the full F1 and ACC tables in Appendix D.3, and the full per-checkpoint sweeps in Appendix D.5.

Table 4.1: Matched horizon (best checkpoint at step ≤ 180), average EM across the three seeds per reward shape.

Reward shape	seed 42	seed 43	seed 44	mean	range
F1+format	0.322	0.292	0.308	0.307	**0.030**
F1-only	0.313	0.283	0.318	0.305	0.035
EM-only	0.301	0.264	0.249	0.271	0.052

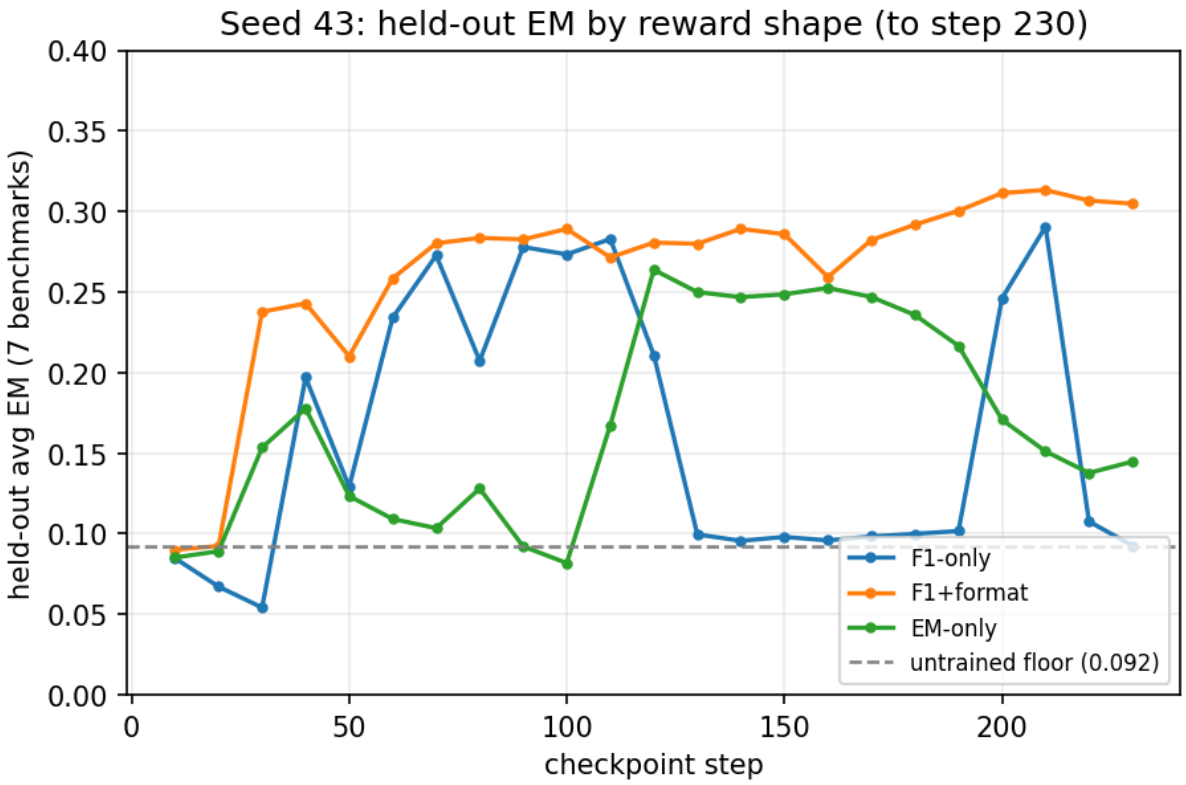


Figure 4.2: Seed 43 held-out EM by reward shape (to step 230); F1+format leads and is the most stable, F1-only is the noisiest with a deep collapse over steps 130 to 190 then a spike at 210 and a crash at 230 and ends the lowest, while EM-only is worst only on the matched-horizon best-checkpoint summary (Figure 4.5) and not on this seed-43 trajectory.

Table 4.2: Best checkpoint at each seed horizon (seed 42 at step $\leq$ 180, seed 43 at step $\leq$ 230, seed 44 at step $\leq$ 312), average EM per reward shape.

Reward shape	seed 42	seed 43	seed 44	mean	range
F1-only	0.313	0.291	**0.352**	0.318	0.061
F1+format	0.322	0.313	0.309	0.315	**0.013**
EM-only	0.301	0.264	0.318	0.294	0.054

Table 4.3: Matched horizon, average F1 per reward shape at each run's best-F1 checkpoint, showing the same ordering on the dense metric.

Reward shape	seed 42	seed 43	seed 44	mean	range
F1-only	0.402	0.369	0.407	0.393	0.038
F1+format	0.413	0.370	0.390	0.391	0.043
EM-only	0.368	0.333	0.319	0.340	0.049

The substring-cover accuracy (ACC) tells the same story on the most lenient metric:

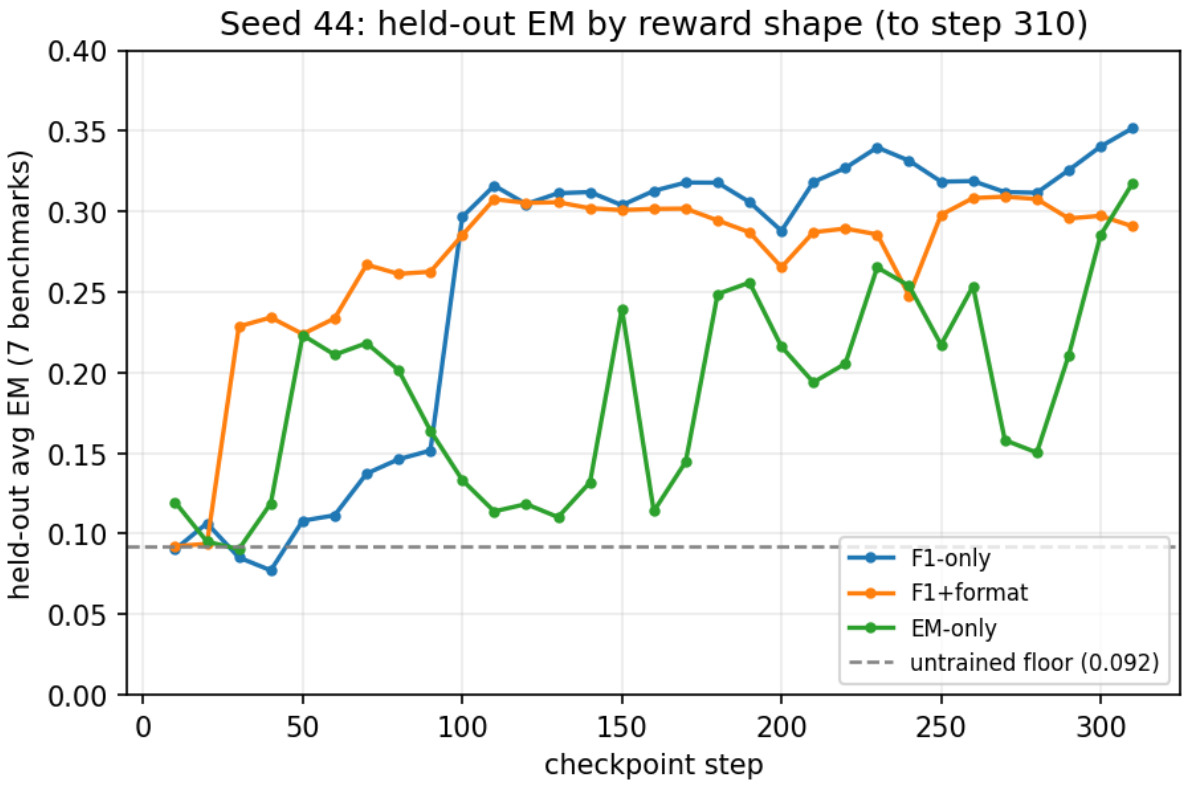


Figure 4.3: Seed 44 held-out EM by reward shape (full epoch, to step 310); F1-only reaches the run-high 0.352 at step 310 and EM-only's late rise is length-driven.

EM-only is worst at every seed here too. ACC is read at each run's seed-horizon best-EM checkpoint.

Table 4.4: Substring-cover accuracy (ACC) at each run's seed-horizon best-EM checkpoint, per reward shape; EM-only is worst on the lenient metric as well.

Reward shape	seed 42	seed 43	seed 44	mean
F1-only	0.394	0.355	0.397	0.382
F1+format	0.378	0.359	0.355	0.364
EM-only	0.334	0.293	0.340	0.323

The full per-run F1 and ACC tables are in Appendix D.3.

4.1.4 What this section establishes (interpretation)

Several findings emerge from the held-out evaluation (Figures 4.1 to 4.6), which we discuss as we present them and map to the research questions at the end of the section.

First, on feasibility, every trained run clears the 0.092 floor by a wide margin (best 0.352); a sub-1B model is trainable with the same system. The same GRPO plus search-tool pipeline that works for 3B-and-larger models also drives a sub-1B model well above its untrained floor on held-out multi-hop QA. The cross-generation comparison in Section 4.2 develops this point.

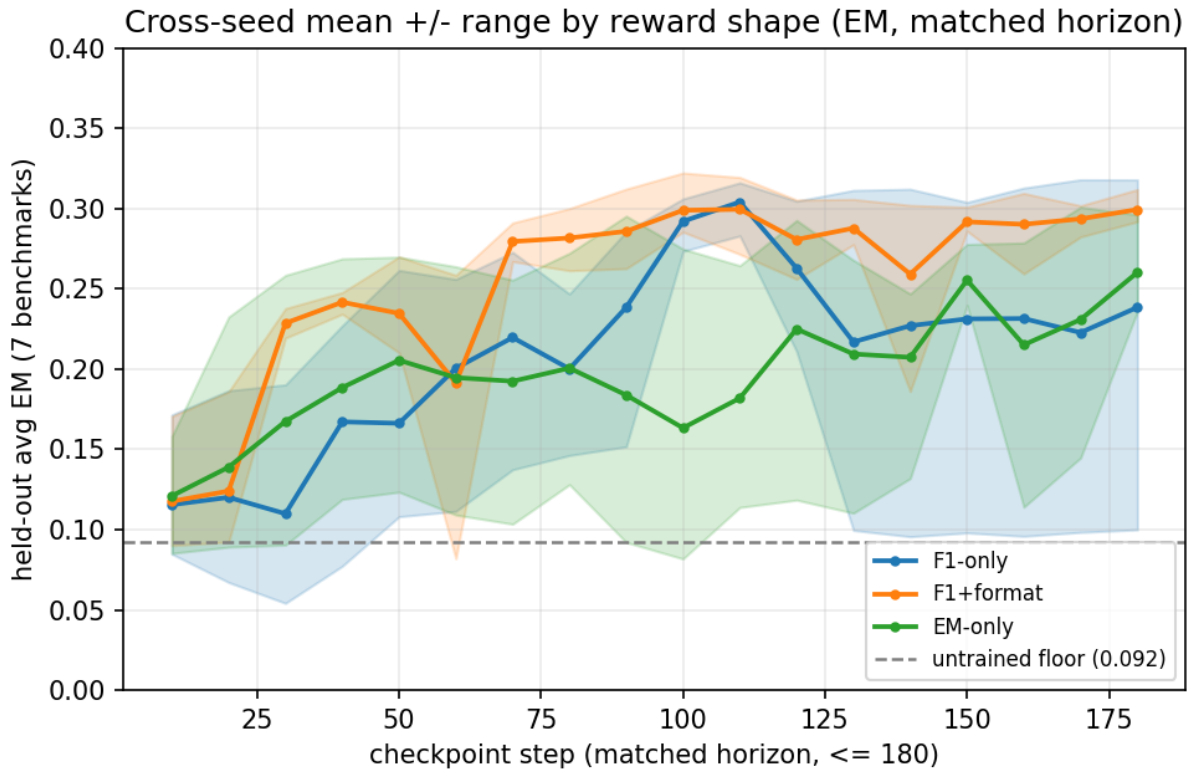


Figure 4.4: Per-reward-shape mean across the three seeds with shaded minimum-to-maximum range, matched to step ≤ 180; the band width is the seed-robustness read, where F1+format is the tightest and most seed-robust while EM-only's mean sits lowest, and both EM-only's band and F1-only's are wide, F1-only's inflated by the seed-43 collapse.

Second, on generalisation across the suite, training is on MuSiQue only, so MuSiQue is the only in-distribution benchmark and the other six (NQ (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), PopQA (Mallen et al., 2023), HotpotQA (Yang et al., 2018), 2Wiki (Ho et al., 2020), Bamboogle (Press et al., 2023)) are strictly out-of-distribution; the gains there are generalisation, not memorisation. This reproduces the ReSearch (Chen et al., 2025) headline that a model trained on MuSiQue alone generalises across benchmarks, and extends it two ways: to a sub-1B model, and to the three single-hop Search-R1 (Jin et al., 2025b) benchmarks (NQ, TriviaQA, PopQA) that ReSearch never evaluated (its four benchmarks are a multi-hop subset of our seven). Per-benchmark best-checkpoint EM and the per-dataset spread are in Appendix D.1 and D.4.

Third, EM-only is the worst reward shape at all three seeds at the matched horizon (0.301 / 0.264 / 0.249) and has the widest seed range (0.052): it is both lowest and least stable. Three seeds are enough for an exact test here: under the null that reward shape has no effect, the three shapes are exchangeable within each seed, and the probability that a pre-specified shape (EM-only, the Search-R1-faithful reward whose transfer the ablation was designed to test) ranks last at all three independent seeds is $(1/3)^3 = 1/27 \approx 0.037$ (test detail and a conservative no-pre-specification variant in Appendix D.7). The one place it looks competitive is at seed 44's full horizon (0.318, edging out F1+format's 0.309), but that is a length effect: seed-44 EM-only ran to step 310, and at the matched step ≤ 180 horizon it is only 0.249. We therefore state EM-only's weakness at the matched

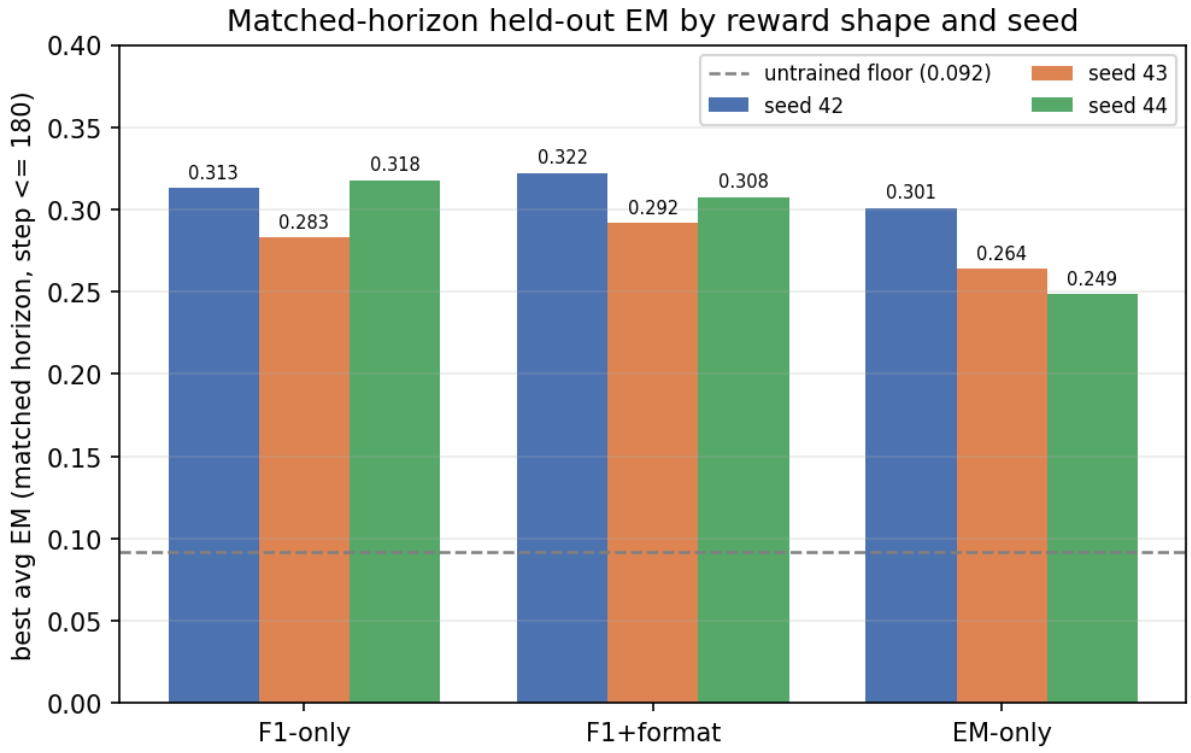


Figure 4.5: Matched-horizon (best checkpoint at step $\leq$ 180) average EM, grouped by reward shape with one bar per seed; this is the compact read of the same data as Figure 4.4, where EM-only is the shortest group at every seed (worst), the per-reward bar spread shows seed variance (F1+format tightest, EM-only widest), and within each group the tallest seed differs because the reward-shape ordering inverts by seed (all seeds are read at the same step $\leq$ 180, so this is not a training-length effect).

horizon, not at the full seed-44 horizon.

The reason is the group-relative advantage that GRPO learns from. The update is driven by the within-group spread of rewards, so when a prompt's rollouts nearly all score alike, that spread (and with it the advantage) collapses; a sparse 0/1 exact-match reward leaves most groups in exactly that regime, whereas a dense token-F1 reward grades every rollout and keeps the spread informative. The training metrics confirm this directly: at the matched horizon EM-only has the lowest gradient norm of the three shapes and is the only shape whose policy entropy stays pinned high rather than sharpening (under-sharpening; clearest at seeds 42 and 43, with seed 44 the exception that does sharpen), as detailed in Section 4.3 and Appendix F. EM-only is worst, then, not by accident but because the sparse reward supplies the weakest learning signal of the three.

Fourth, the ordering is metric-robust across EM, F1, and ACC (Figure 4.6). EM-only is worst on F1 at every seed (matched F1 0.368 / 0.333 / 0.319 versus roughly 0.37 to 0.41 for the F1 rewards) and worst on ACC at every seed (mean ACC 0.323 versus F1-only 0.382 and F1+format 0.364); its F1 deficit is in fact larger than its EM deficit, since it optimises exact match rather than partial overlap. So "EM-only is worst" is a property of the policy, not an artifact of scoring on the same EM metric it optimises: the ranking holds on two metrics it was never trained toward.

Fifth, the reward-shape ordering inverts by seed: F1+format wins seeds 42 and 43 (on both the matched-horizon and the seed-horizon view), while F1-only wins seed 44. No

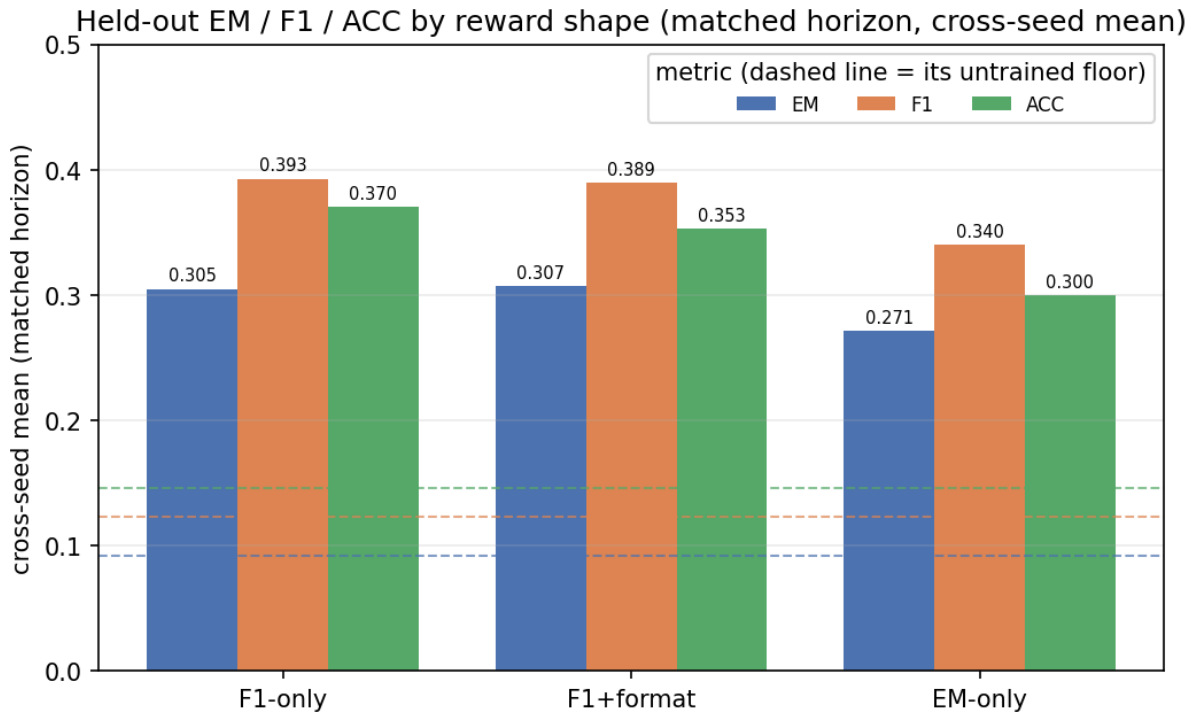


Figure 4.6: Matched-horizon (step $\leq$ 180) cross-seed mean of EM, F1, and ACC at each run's best-EM checkpoint, by reward shape; dashed lines mark each metric's untrained floor. EM-only is lowest on all three metrics; the two F1 shapes are close and higher. The per-run numbers are in Appendix D.6.

single reward shape dominates all seeds, so we report the ranking as seed-dependent; a 2-to-1 seed split carries no statistical signal at $n = 3$ (Appendix D.7).

Sixth, F1+format is the most seed-robust (tightest range: 0.013 at the seed horizon, 0.030 at the matched horizon), while F1-only has the highest ceiling (0.352 at seed 44, step 310). This is the practical trade-off: F1+format for reliability, F1-only to chase the ceiling (at the cost of more training length). One clarification keeps two senses of "stable" apart. The seed-robustness just described is a spread across the three seeds' best checkpoints, and on that measure F1+format wins. A different quantity, cadence-to-cadence trajectory stability (the mean step-to-step change in held-out EM along a single run), ranks mostly by seed rather than by reward shape: the three calmest runs are seed-44 F1+format (0.015), seed-44 F1-only (0.016), and seed-43 F1+format (0.017), while the seed-42 F1+format leg is near the worst (0.051, an isolated outlier step), and EM-only's runs rank fourth, sixth, and seventh on top of a secondary saw-tooth. So F1+format is the most seed-robust on the metric that matters for picking a recipe (best-checkpoint spread); it should not be over-read as the calmest trajectory. The full nine-run cadence-stability table is in Appendix F.1.

Seventh, on why F1+format edges F1-only at seed 42, the drivers are format-adherence and retrieval-grounding, not chain coherence. By format-adherence we mean the fraction of rollouts that end in a parseable `<answer>` block, and by retrieval-grounding the fraction of answer tokens that appear in the retrieved passages; chain coherence is whether the reasoning steps actually support the final answer, measured by the LLM-judge chain-flip audit (Appendix J). Decomposing the +0.9 percentage-point F1+format-over-F1-only lift on the seed-42 best checkpoints: the valid-answer rate rises +8.9 percentage points (89.3

% to 98.2 %) and answer-token retrieval-grounding rises +5.5 percentage points (0.830 to 0.885), each far larger than the +0.9 point EM lift they produce, while silent chain-flips do not fall (they are, if anything, slightly more frequent for F1+format). So the format gate helps by pulling the policy toward emitting a well-formed, retrieval-grounded answer, which is upstream of getting EM right, rather than by producing more coherent reasoning chains; the chain-flip audit detail is in Appendix J. Read across all three shapes, the same valid-answer lens shows EM-only's deficit is a termination gap rather than a per-search answer-quality gap: at a matched completed search count the three shapes answer about equally well, but EM-only aborts far more held-out rollouts (Appendix D.8).

Eighth, on significance with three seeds, the reward-shape gaps smaller than the cross-seed range are within seed noise: F1+format minus F1-only at the matched horizon is +0.9 / +0.9 / -1.0 percentage points across seeds 42, 43, and 44 (mean +0.3 percentage points, and the sign flips), so that ordering is seed-dependent, not a reliable win. The one robust ordering is EM-only last: both F1 shapes beat it at every seed (F1-only by +1.2 / +1.9 / +6.9 percentage points). Three seeds is too few for a powered significance test; we state the seed-count limit in Limitations 5.1.

Ninth, the 0.1 floor and format gate do not mask the tool-use signal at this scale; they aid discovery. This inverts the pilot finding at 0.6B, where the floor masked the signal. The pilot result and its figure are in Appendix H.1; the floor-drop motivation is introduced in Method 3.4; the mechanism (format-adherence plus retrieval-grounding) is the seventh point above (quantified) and is developed in Section 4.3.

These findings map to the research questions in Introduction 1.2 (RQ1 feasibility and generalisation; RQ2 best reward shape; RQ3 most stable reward shape); the RQ-level synthesis is given in the Conclusions.

Full held-out detail (the per-benchmark best-checkpoint table, exact per-leg end-steps, per-dataset best-checkpoint spread, F1 and ACC tables, and links to all per-checkpoint sweeps) is in Appendix D.

4.2 Cross-generation comparison

The headline lift is from an untrained 0.092 to a best trained 0.352, an absolute gain of +0.260.

Before reading that lift, we validate the eval network it rides on. The Search-R1 Qwen2.5-3B GRPO checkpoints reproduce within ±2.5 percentage points on this same eval network (base -2.4 percentage points, instruct +0.5 percentage points; from our full-data reproduction), which validates the pipeline the 0.8B numbers depend on. The full paper-versus-ours per-benchmark table (paper base and instruct versus our reproduced base and instruct) is in Appendix E.1.

The cross-generation anchor is the floor-versus-trained comparison on that same network.

Table 4.5: Floor-versus-trained average EM on the same eval network for the older-generation Qwen2.5-3B and the current-generation Qwen3.5-0.8B, with the Reinforcement Learning (RL) lift for each.

Model	Untrained floor	Trained	RL lift
Qwen2.5-3B (older gen)	0.199	0.341	+0.142
Qwen3.5-0.8B (current gen)	0.092	**0.352**	**+0.260**

Several caveats are required to read this comparison honestly. Size is confounded with model generation, and the two trained endpoints were not produced by the same training run: our Qwen3.5-0.8B was GRPO-trained on MuSiQue with the reward-shape ablation here, whereas the Search-R1 Qwen2.5-3B was trained by the original authors on NQ and HotpotQA with the EM-only reward (we evaluate both on the identical eval network and EM metric, but the data and reward differ). The defensible claim is therefore that Reinforcement Learning (RL) helps the small newer-generation model most, not that our recipe beats the paper's. We also anchor to our own 3B reproduction (0.341), not the paper row. We frame the result as a capability frontier moving down the parameter curve per generation. The full floor-plus-trained provenance table is in Appendix E.2.

A training-exposure asymmetry strengthens the claim. The 0.8B endpoint reaches 0.352 on roughly one twenty-sixth the released Search-R1 checkpoint's prompt exposure (19,968 prompts, one MuSiQue epoch, versus the released 3B checkpoint's 514,560 prompts at about three epochs over the NQ-plus-HotpotQA corpus, which is 8.5 times larger) and about half of ReSearch's MuSiQue exposure (one epoch versus two). So, at its best checkpoint, the small newer-generation model reaches held-out exact-match comparable to the larger one despite far less training, which makes the cross-generation reading more, not less, conservative. (The released 3B checkpoint trained 1005 steps, not the paper-text 500: a reproduction-integrity detail in Appendix E.)

The floor-versus-trained numbers above are the empirical answer to the secondary "post-training versus parameter count" question (RQ5). We treat RQ5 as discussion-framed rather than a primary result: the full problem statement, the research question, and the bundle confound (model generation, post-training, and recipe are entangled) are set out in Introduction 1.2, and the controlled same-generation follow-up that would disentangle them is proposed in Limitations and Future Work (Chapter 5). We do not restate that framing here.

4.3 Training dynamics and emergent compression (mechanism)

This section explains the Section 4.1 shapes from the per-step GRPO rollout metrics. Every line is the centred moving average drawn over the faint raw per-step series, so the trend and the background noise (the stability signal) are both visible. Health diagnostics (gradient norm, loss, KL divergence, entropy, termination) are in Appendix F.

4.3.1 Per-seed trajectories

For each seed, the three reward shapes are overlaid on the five body metrics: reward (top-left), tool calls per sample (top-middle), response length in tokens (top-right), total tokens per sample (bottom-left), and length-clip ratio (bottom-middle). Each of Figures 4.7 to 4.9 shows one seed. The signature shared by all three is that reward climbs while the four cost panels spike in the first 10 to 20 steps (the cold-start over-search) and then fall (compression); what differs between seeds is which reward shape breaks that pattern, and on which panel.

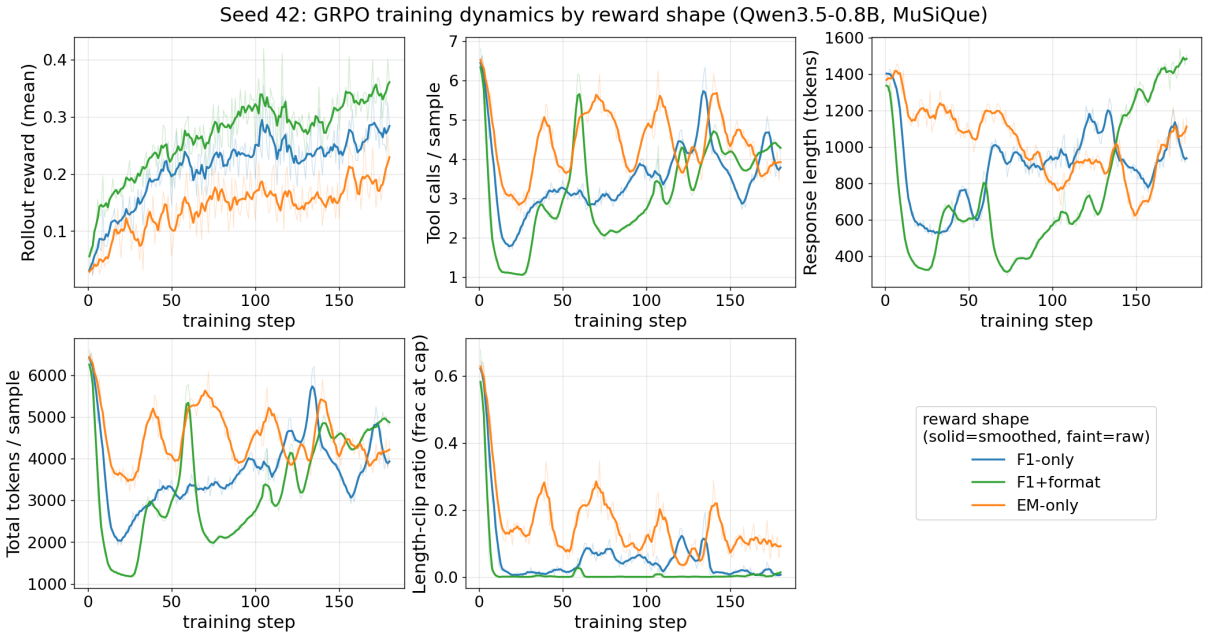


Figure 4.7: Seed 42 training dynamics, capped at the standard horizon step 180 (per-seed caps are 180 / 230 / 312 for seeds 42 / 43 / 44, with the last full cadence at step 310).

At seed 42 (Figure 4.7), F1+format has the highest, still-rising reward at the step-180 cap; F1-only climbs smoothly to about 0.29; and EM-only is lowest (plateau around 0.22). All three cut tool calls from about 6.5 toward about 4. The shapes split on the length panels: F1-only and EM-only shorten responses (F1-only from about 1390 to about 950 tokens) while F1+format's response length rises (from about 1360 to about 1500 tokens) even as its calls and clip ratio fall (the "long chains, few calls" regime). Length-clip collapses to about 0 for both F1 shapes, but EM-only stays off the floor (about 0.10), the first sign it is still hitting the 8192-token cap.

At seed 43 (Figure 4.8), F1+format again peaks highest (about 0.36 near step 134) but drifts down over the back half; F1-only is the noisy leg, with reward peaking about 0.31 near step 177 then falling to about 0.24 by step 230, the training-side echo of its held-out collapse in Figure 4.2. F1+format compresses tool calls the hardest of any run (from about 6.8 to about 2.8). Here the exception flips to the length panel: EM-only's response length rises (from about 1350 to about 1560 tokens), a visible over-search re-entry, and its clip ratio again stays elevated (about 0.14) while both F1 shapes sit at about 0.

At seed 44 (Figure 4.9, the only full-epoch leg) we see the textbook compression view: for both F1 shapes, reward climbs while tool calls, response length, total tokens, and clip ratio all fall together (the precise cold-start-to-end magnitudes are in Results 4.3.4). EM-only is the counter-example, and the full epoch is what exposes it. Like the F1 shapes,

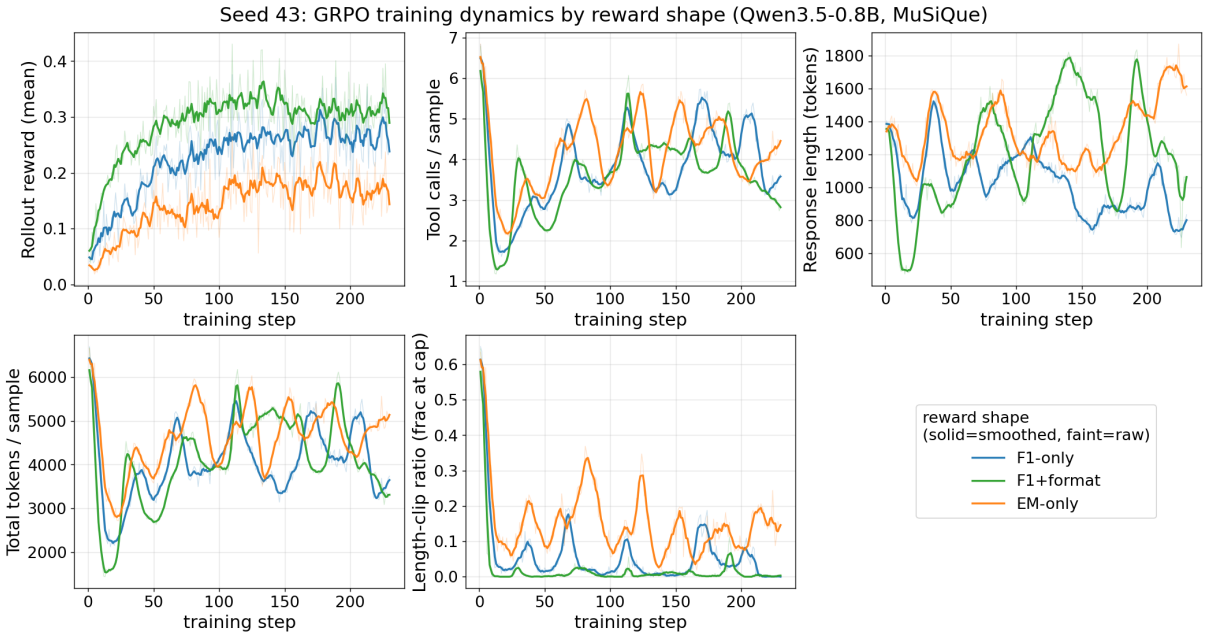


Figure 4.8: Seed 43 training dynamics, capped at the standard horizon step 230.

EM-only leans early (tool calls drop from about 6.5 toward 4 by mid-horizon, the same window the shorter seed-42 and seed-43 legs stop in), but unlike them it does not settle: over the back half it re-enters over-search and its tool-call count climbs back to 6.2 by step 310, near its 6.5 cold start. Its length-clip tracks the same failure, rising back to about 0.36 at the end (still hitting the 8192-token cap on over a third of rollouts, versus about 0 for the F1 shapes), while its reward stays lowest. This is not an artifact of where we stopped: it is the explore-and-lean oscillation failing to converge, visible only because seed 44 ran a full epoch, and it is the mechanism behind EM-only's worse held-out EM in Section 4.1.

The reward panel reveals three reward-trajectory archetypes across seeds. F1-only shows a slow ramp and a smooth climb with no catastrophic collapses. F1+format shows a medium-fast ramp and the highest peak (its seed-42 reward tops out near 0.42), with wide single-cadence swings that recover within a cadence (high single-cadence variance around a tight cross-seed mean). EM-only shows the fastest early ramp and then a flat, low plateau with no recovery mechanism; its held-out exact-match accordingly peaks early and then slips (Section 4.1), which is the mechanism behind its low matched-horizon EM and its instability.

What this means is that the F1 shapes turn a dense outcome reward into shorter, cheaper chains at equal-or-better reward (emergent compression, detailed in Results 4.3.4), while EM-only's sparse 0/1 reward gives a weak gradient to trim a chain that already produces the answer, so it leans only briefly and then drifts back to the length cap, which is why it is both lowest and least stable in Section 4.1.

4.3.2 Cross-seed view (seed-robustness of the dynamics)

For each reward shape, we overlay seeds 42, 43, and 44 on reward, tool calls, response length, and length-clip (Figures 4.10 to 4.12). The trend is seed-stable: the same cold-start collapse plus plateau at every seed, gradient norm bounded throughout (Appendix F),

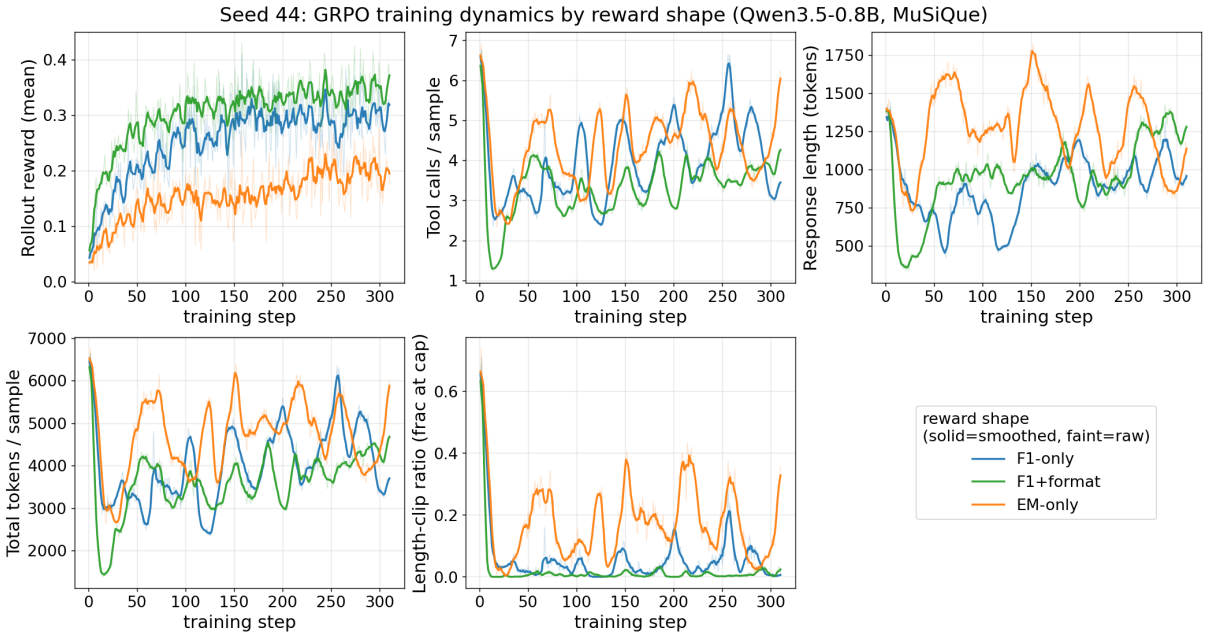


Figure 4.9: Seed 44 training dynamics (full epoch; standard horizon 312, last full cadence step 310); this is the clearest view, with reward climbing while tool calls, length, total tokens, and clip ratio all fall for the F1 shapes while EM-only stays high.

and no divergence.

4.3.3 Cross-seed average (matched horizon)

Figure 4.13 collapses all three seeds into one cross-seed mean per reward shape (with a minimum-to-maximum band), matched to step ≤ 180, for reward, tool calls, and response length. It is the compact read of the whole section: averaged across seeds, the F1 shapes lean (falling tool calls and length) while reward climbs, whereas EM-only stays highest on tool calls and lowest on reward.

4.3.4 Emergent compression (key insight)

Under the unmodified outcome reward, the F1 shapes compress from a verbose, over-searching cold start into a lean tool-caller; EM-only is the exception. The seed-44 full-epoch cold-start-to-end magnitudes (step 1 to step 310) make this concrete. F1-only cuts tool calls from 6.6 to 3.4, response length from 1335 to 962 tokens, total tokens from 6435 to 3694 (cut by about two-fifths), length-clip from 0.63 to 0.01, and aborted from 0.72 to 0.01. F1+format cuts tool calls from 6.5 to 4.3, total tokens from 6360 to 4730, length-clip from 0.63 to 0.01, and aborted from 0.72 to 0.01 (the cleanest output discipline). EM-only, the exception, barely moves: tool calls end at 6.2 (from 6.5, after leaning mid-horizon and re-entering over-search), total tokens fall only 6380 to 6031, length-clip stays high (0.62 to 0.36) and aborted with it (0.72 to 0.37). Our sub-1B F1 runs show search-call count and total tokens fall during RL while held-out EM rises. Per-rollout response length itself is mixed: it falls for F1-only but rises for F1+format at seed 42 (and for EM-only at seed 43), and in every run it dips early then partly rebounds, the same decrease-then-increase shape Search-R1 reports; the net token reduction is therefore driven by fewer searches, not by shorter reasoning. So at small scale the search-and-token budget shrinks where the

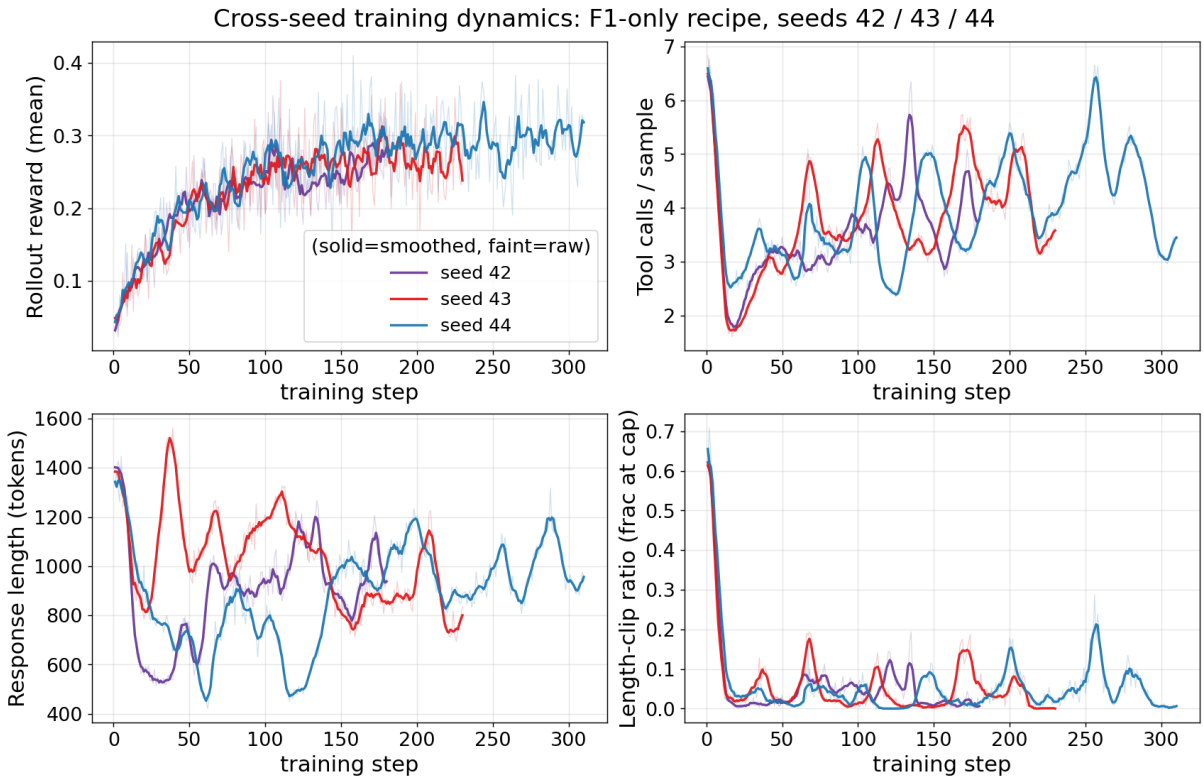


Figure 4.10: F1-only across seeds 42, 43, and 44; the cold-start collapse-then-climb in reward and the early spike-then-fall in tool calls and length repeat at every seed, seed 43 is the visibly noisiest leg (its back-half reward drift), and seeds 42 and 44 track each other closely over the shared horizon.

larger-scale dynamic has it grow. EM-only's failure to compress is the mechanism behind its worse held-out EM (Section 4.1) and its instability.

Why do chains shorten without an explicit length penalty? The reward has no length term, so the compression is emergent. The likeliest driver is the 8192-token generation cap acting as an implicit length penalty: a rollout that runs to the cap never closes its `<answer>`, so it scores 0. The evidence is two within-run correlations, each a Pearson correlation computed over a run's per-step training series. First, the length-clip ratio (fraction of rollouts truncated at the cap) and the no-valid-answer (aborted) ratio move together almost perfectly (Pearson +0.98 to +0.99 in every run), so hitting the cap and failing to produce a scorable answer are effectively the same event. At cold start 62 to 68 % of rollouts hit the cap (and about 72 to 75 % emit no valid answer), so a large fraction of the cold-start population scores 0 purely for being too long. That places a strong group-relative advantage on the shorter, finished rollouts. Second, reward anti-correlates with the clip ratio (-0.44 to -0.52 for the F1 shapes), so escaping the cap is the main early route to reward. As the policy learns to finish before the cap, the clip ratio falls to about 0 and chains shorten, with no length term ever written into the reward.

This also explains why EM-only is the exception. The cap penalty bites only through "did the rollout finish with a rewarded answer". F1's dense partial credit rewards any finished answer with token overlap, so "finish before the cap" is almost always worth points and the shortening gradient is strong and persistent. EM-only's 0/1 reward pays a finished answer only when it is exactly right (rare at this scale), so a finished-but-wrong rollout scores 0, the same as a capped one; the gradient separating "finish" from "hit the

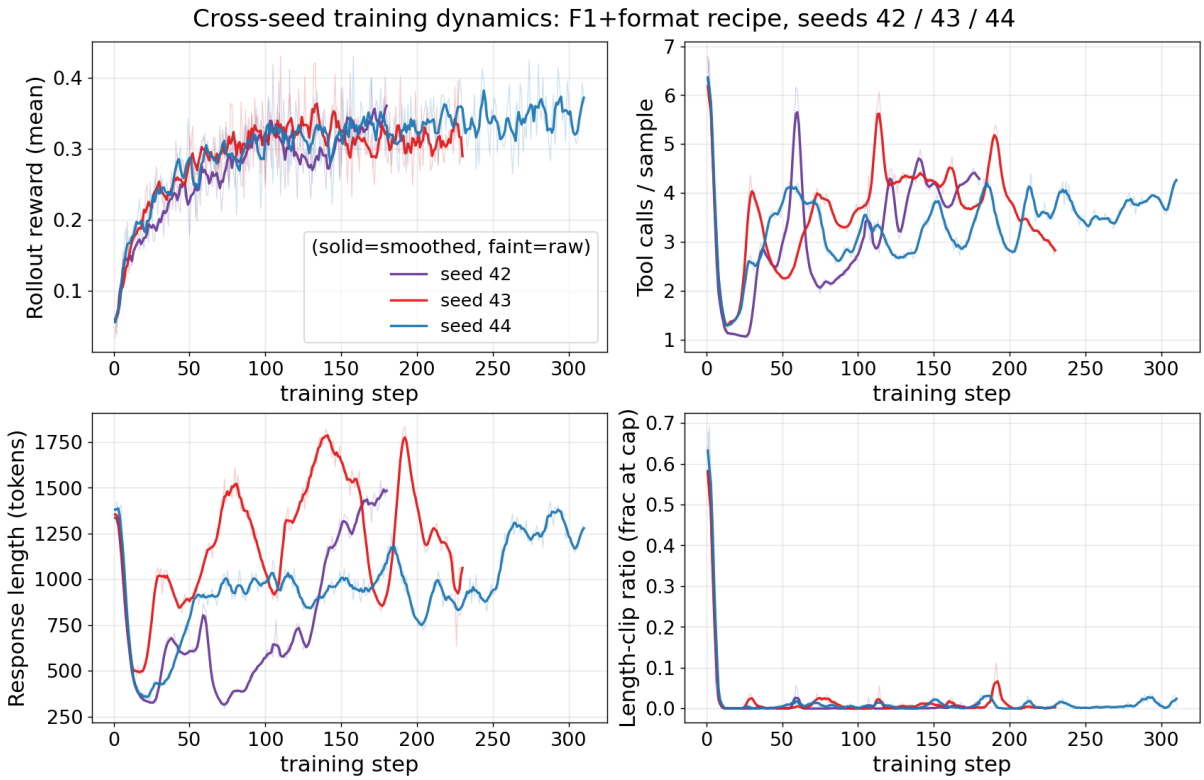


Figure 4.11: F1+format across seeds 42, 43, and 44; this is the tightest cross-seed reward band of the three shapes (the seed-robustness read of Figure 4.4), the single-cadence dips are seed-local and recover within a cadence, and all three seeds settle into the same low-call, longer-chain regime.

cap" is muted. The same reward-versus-clip-ratio correlation that is -0.44 for F1-only at seed 44 is only -0.20 for EM-only at that seed, i.e. for EM-only the reward barely rewards escaping the cap, so the shortening pressure is weak. Consistently, EM-only keeps roughly 10 to 36 % of rollouts at the cap at the last full cadence (9.7 % at seed 42, 14.1 % at seed 43, 35.6 % at the full-epoch seed 44; versus about 1 % for the F1 shapes). A secondary, untested contributor is the model prior: Qwen3.5's post-training favours concise tool use, so once the cap gradient permits, RL may be collapsing into the model's existing concise style rather than discovering a new one; a base-model (no post-training) run would separate the cap effect from the prior, and is left to future work (Section 5.2).

Compression is a cost property, not an accuracy lever, and it is important not to over-read it. Being leaner does not make a checkpoint more accurate: across the nine runs the best-EM checkpoint sits mid-to-high on training-time tool-call count (for F1-only seed 43 it is the twenty-second-leanest of 23 checkpoints; for EM-only seed 44 it is the least lean), and within a run, held-out EM correlates positively with tool-call count in 7 of the 9 runs (about +0.45 for all three F1-only runs; only the two seed-42 non-F1-only runs go slightly negative), i.e. the checkpoints that search a little more tend to score a little higher. So the right reading of compression is "fewer searches and tokens at equal-or-better accuracy" (a deployment-cost win), not "leaner means more accurate". The same split shows up at evaluation, where EM-only aborts far more held-out rollouts than the F1 shapes at every best checkpoint (Appendix D.8).

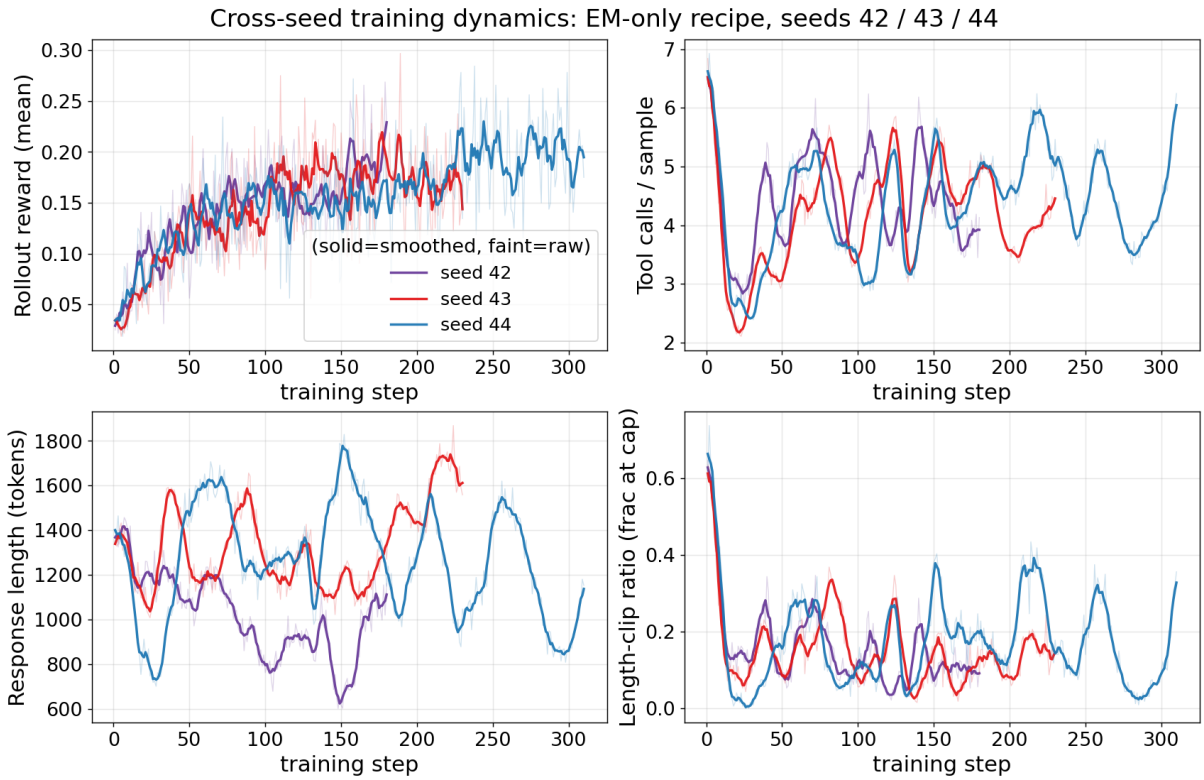


Figure 4.12: EM-only across seeds 42, 43, and 44; the late over-search re-entry is visible only at the full seed-44 horizon.

4.3.5 From cold start to lean to planning: three rollouts

What do the rollouts look like as they go from cold start to lean to planning? Three shortened rollouts from the F1-only seed-42 run make the compression concrete (full traces in Appendix G). At cold start (step 1, reward 0) the format is already perfect but entity resolution is not. About 93 % of cold-start rollouts run searches until they approach the 8192-token cap; this one instead commits early: on "what tradition is the basis for the legal system in the place where Coloane Park is located?" the policy never resolves Coloane to Macau and, after a single search, confabulates a non-existent answer, `<answer> Kissing Traditions </answer>`. By the first lean (around step 21, reward 1.0), on "who won season 5 of Bigg Boss presented in the language of the novel *Aavarana*?", the policy resolves *Aavarana* to Kannada to Bigg Boss Kannada to the winner in 3 searches and about 6.6 thousand characters, emitting `<answer> Chandan Shetty </answer>`. (A sibling rollout that skips the language bridge defaults to the Hindi show and fails: the persistent bridge-entity brittleness.) By later planning (around step 39, reward 1.0), on "how many floors are in the Trump Tower in the city where the performer of *Come 2 My House* was born?", the policy writes a numbered plan, issues one query per hop (to Chaka Khan, then born in Chicago, then Chicago Trump Tower), and resolves the Chicago-versus-New-York ambiguity from chain context, emitting `<answer> 98 </answer>`.

The higher-level significance, that the search-and-token budget shrinks where the prior papers report it growing and that a leaner agent at equal-or-better accuracy is the practical takeaway, is taken up again in the Conclusions (Chapter 6).

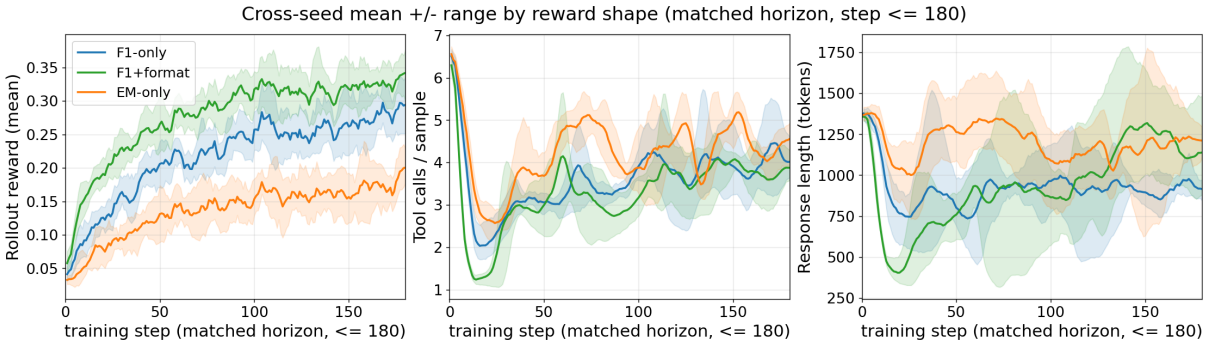


Figure 4.13: Cross-seed mean with minimum-to-maximum range per reward shape, matched to step $\leq$ 180, for reward, tool calls, and response length; this compact compression-and-reward read shows the F1 shapes leaning while reward climbs, while EM-only stays highest on tool calls and lowest on reward.

4.3.6 Higher reward is not more exact matches: the 20 to 22 % training-time EM ceiling

A reward shape can raise the optimisation scalar without producing more correct answers. The following table reports training-side run-high reward versus run-high exact-match rate (%R=1, the fraction of rollouts that score a 0/1 exact match; canonical n=3200-cadence values, seed 44).

Table 4.6: Training-side run-high reward versus run-high exact-match rate (%R=1) at seed 44 for the F1-only, F1+format, and EM-only runs.

Reward shape	Reward	Run-high reward (cadence-mean, n=3200)	Run-high %R=1 (exact-match rate)
F1-only	F1	0.314 (C30)	**21.9 %** (C25)
F1+format	F1 + 0.1 format	**0.358** (C30)	20.5 % (C28)
EM-only	EM	0.216 (C28)	21.6 % (C28)

There are two reads, both answering RQ6 ("does a higher training reward actually correspond to a higher exact-match rate?"). Within the scalar, the answer is no: the F1+format run posts the highest reward (0.358) but the lowest %R=1 (20.5 %), because the 0.1 format floor pays well-formed wrong answers and inflates the scalar without adding exact matches. The training reward is therefore not comparable across shapes (this is why Section 4.1 ranks on held-out EM, not training reward). The same gap is even cleaner at cold start, before behaviour diverges: at the first cadence (C1) all three shapes are tied on exact match (%R=1 between 4.5 and 4.8 %), yet the reward scalar already reads 0.045 (EM-only), 0.072 (F1-only), and 0.113 (F1+format), so the floor and partial credit inflate F1+format's reward to about 2.5 times EM-only's purely from answer formatting, with zero exact-match difference. There is also a shared ceiling: all three shapes peak at a 20 to 22 % training-time exact-match rate (cross-shape spread only 1.4 percentage points; the full C1 to C31 per-cadence triad at seed 44 is in Appendix F.2), mirrored on held-out data where the seed-44 runs top out around average EM 0.30 to 0.35 (Section 4.1). So 20

to 22 % is a property of the model-and-data scale, not of the reward shape: the reward shape moves held-out EM by a few points within that ceiling; it does not break through it. The structural reason is traced in Results 4.3.4 and Appendix G (bridge-entity brittleness under noisy retrieval, which no reward shape here fixes).

4.3.7 Oscillations (explore-and-lean)

Reward and behaviour oscillate rather than converge monotonically: the policy cycles between an explore phase (more searches, longer chains, entropy rebounds) and a lean phase (fewer searches, reward peaks). Reward peaks then drifts (at seed 44: F1-only from a raw peak 0.410 at step 158 to an end of 0.320; F1+format from 0.430 at step 158 to 0.367), and EM-only's late over-search re-entry (seed 44, end at 6.2 calls) is this oscillation failing to settle. It shows as the saw-tooth in the raw series behind the exponential moving average (Figures 4.7 to 4.9) and the roughly-four-cadence-cycle peaks noted for F1+format. This is why checkpoint selection matters (Results 4.1.3) and why we report per-cadence eval, not just the endpoint.

4.3.8 What the joint read establishes (and what is missing)

On stability, the gradient norm stays bounded at every seed (no divergence); the only total collapse was a tensor-parallel run (entropy collapse plus gradient-norm explosion under the same recipe and seed, so the tensor-parallel setting is implicated), detailed in Appendix F.

On the health gestalt (Appendix F), entropy declines for five of the six F1 legs (sharpening) but stays pinned high for EM-only at seeds 42 and 43 (under-sharpening: a sparse 0/1 reward gives a weak gradient, hence a low gradient norm, so the policy never commits; seed-44 EM-only is the exception that does sharpen); KL divergence rises modestly (beta of 1e-3); and the loss is decoupled from reward (as expected under GRPO).

On termination fate, natural-termination is approximately one minus length-clip in our runs (correlation 0.93 to 0.99), and aborts are overwhelmingly truncations (the non-truncation residual is at most about 2 % of rollouts on run average), so the decisive signal is EM-only's failure to compress, not a separate malformed-output mode.

A still-climbing caveat applies: several runs were length-bounded (still rising at the end), so the endpoint numbers are conservative lower bounds.

Finally, on what is not yet logged (future work): per-turn retrieval-hit and search-quality metrics would separate "leaner because better-targeted search" from "leaner because fewer searches".

The full training-health diagnostics are in Appendix F.

5 Limitations and Future Work

5.1 Limitations

Several scoping decisions bound the conclusions we can draw, and we state them plainly here before discussing the work they motivate.

Our results rest on a single model family and size. Every trained checkpoint uses Qwen3.5-0.8B (Qwen Team, 2026); we provide no cross-architecture or cross-scale validation, so the reward-shape findings should be read as conclusions about one small model rather than about Retrieval-Augmented Reinforcement Learning in general.

Training was likewise confined to a single dataset. We trained on MuSiQue (Trivedi et al., 2022) only, chosen as the hardest of the multi-hop sets, and we measure generalisation held-out across the seven-benchmark suite. Single-dataset training is a deliberate and precedented choice rather than an oversight: ReSearch (Chen et al., 2025) likewise trains on MuSiQue alone and still generalises across benchmarks, and we evaluate held-out on seven benchmarks to its four. What we do not do is ablate training-data diversity, so we cannot separate the effect of the reward shape from the effect of the specific training distribution; broadening the training mix is left to future work.

The training horizon is also limited. Most runs were length-capped at roughly one epoch or earlier, and several were still improving at the point we stopped them; the headline numbers for those runs are therefore conservative lower bounds rather than converged values. The length-normalisation analysis is the appropriate lens for these cases, because it compares runs at matched horizons rather than at their (unequal) stopping points.

Our seed count is small. We ran three seeds per reward shape, which is enough to surface a seed-by-reward-shape interaction but not enough to give tight variance estimates. The headline last-place ranking is the one claim strong enough to clear an exact test at this seed count (EM-only last at all three seeds, $p \approx 0.037$; Appendix D.7), but the pairwise orderings are not: the seed-44 inversion of the reward-shape ranking is left as an open question rather than resolved, since three seeds cannot distinguish a genuine interaction from sampling noise with confidence.

We report only automatic surface metrics. The main results use exact-match (EM), token-level F1, and substring accuracy, with no human or large language model (LLM) judge quality scoring; the chain-flip LLM-judge audit is a separate, scoped study and does not feed the headline numbers.

Finally, performance is ceiling-bound by retrieval and the 0.8B prior, not by the reward. All three reward shapes plateau at a roughly 20 to 22% training-time exact-match rate (Results 4.3.6), and the dominant failure mode across all rewards is bridge-entity brittleness under noisy retrieval (Appendix G). The reward-shape ablation moves held-out EM only *within* that ceiling, so reward design is not the lever that breaks it; the headline gains are bounded by the retriever and the model prior rather than by the choice among F1-only, F1+format, and EM-only.

5.2 Future Work

The limitations above point to a coherent set of follow-up experiments, which we sketch here in roughly descending order of how directly they would sharpen the present conclusions.

The most important follow-up is a same-generation control that separates the share of the cross-generation result attributable to model generation from the share attributable to parameter count. The within-model effect of Reinforcement Learning with Verifiable Rewards (RLVR) at sub-1B scale is already in hand: the untrained-versus-trained Qwen3.5-0.8B pair of Section 4.2 is a +0.260 exact-match lift on one model and one evaluation network. What the RQ5 reading bundles together is the newer generation and the parameter count, and separating those needs a capability-frontier control in the same family: an untrained larger model (for example Qwen3.5-2B) set against the RLVR-trained Qwen3.5-0.8B, or RLVR applied to that larger same-generation model. This is exactly the comparison the bundled confound around RQ5 cannot make (Introduction 1.2 and Section 4.2), and it is the controlled follow-up that both of those sections defer to here.

A larger Qwen3.5 run (2B or above) would test whether our scale-dependent findings survive at larger capacity. Two effects are at stake: whether the "EM-only is worst at all seeds" result and the partial-credit-floor effect persist, or whether both invert with scale. Answering this would tell us how far the small-model conclusions transfer.

A semantic-similarity reward is the natural next point on the reward-density axis introduced in Method 3.4. Here we would replace or augment token-level F1 with a meaning-level signal: an embedding-cosine score (Reimers and Gurevych, 2019) or a BERTScore-style score (Zhang et al., 2020), so that paraphrases and synonyms of the gold answer earn graded credit that exact-match and token-F1 both miss. The finding that the denser F1 reward beats sparse EM-only at this scale (Section 4.1) motivates testing whether the density trend continues at the meaning level, or whether a continuous learned-similarity reward instead invites over-optimisation and reward-hacking (the "tricks may hurt" risk) and drifts away from the EM/F1 protocol the field reports on.

Extending the held-out suite with an LLM-judge or faithfulness metric beyond EM and F1 would address the metric limitation above, giving a quality signal that surface-overlap scores cannot capture.

Targeting the roughly 20 to 22% ceiling directly is the final and arguably most consequential direction. Because Results 4.3.6 and Appendix G locate the bottleneck at entity resolution under noisy retrieval rather than at the reward, the productive levers are on the retrieval and decoding side: stronger or exact retrieval (the flat index, reranking, a larger top-k) together with bridge-entity-aware decoding. Running these would test whether the ceiling is retrieval-bound or prior-bound, which is the question the reward-shape ablation alone cannot settle.

6 Conclusions

6.1 Feasibility verdict

This thesis set out to teach a small language model, one well under a billion parameters, to answer hard multi-step questions by searching for evidence and reasoning over what it finds, training it only on whether its final answer was correct. The first half of our main research question, together with RQ1, asks whether RL with verifiable rewards over a search tool can be made to work on a small, current-generation model under a realistic compute budget. The answer is yes. A sub-1B current-generation model (Qwen3.5-0.8B (Qwen Team, 2026), hybrid) can be GRPO-trained (Shao et al., 2024) with an interleaved search tool, for about one epoch on MuSiQue (Trivedi et al., 2022), to clear its untrained floor decisively on held-out multi-hop question answering. Across the 7-benchmark suite, average exact match (EM) rises from **0.092** to **0.352** (+0.260, roughly a 3.8 times improvement). The 0.352 headline is a conservative lower bound rather than a converged ceiling: the best run (seed-44, F1-only) was still climbing at the budget horizon (see Results 4.3 and Appendix F).

The gain reflects generalisation rather than memorisation. Six of the seven benchmarks are strictly out-of-distribution, and MuSiQue is the only in-distribution dataset entering the average; the model therefore improves on held-out evaluation suites it never saw during training. This settles the feasibility question on its own terms: under a modest compute budget (a single rented multi-GPU node, about $120 to $140 per full-epoch run), an unmodified outcome reward, and a single training corpus, the model learns a transferable reason-over-search policy rather than a lookup table over MuSiQue. The "which reward shape is best" half of the main question is answered below; the cross-generation reading (RQ5) follows in the final section.

6.2 The optimised-recipe answer

The second half of the main question, spanning RQ2, RQ3, and RQ6, asks which reward shape is best and what we learn from it. Our results yield a recipe that is concrete and reproducible, together with a clear negative result on the Search-R1-faithful EM-only reward and a finding about how small models adapt under an unmodified outcome reward.

Reward shape matters, and the Search-R1-faithful (Jin et al., 2025b) EM-only reward is the wrong default at this scale. EM-only is the worst of the three reward shapes at all three seeds at matched horizons (an outcome with probability $1/27 \approx 0.037$ under a no-effect null; Section 4.1), and it is worst even on the EM metric it directly optimises. This is the headline negative result of the ablation: importing Search-R1's sparse 0/1 reward into a sub-1B regime does not transfer.

Within the two partial-credit shapes, the ordering is genuinely seed-dependent and we report it as such rather than collapsing it to a single verdict. F1 combined with a 0.1 floor and a format-gate (F1+format) is the most seed-robust, modal winner: it wins

seeds 42 and 43 and posts the tightest cross-seed spread. F1-only has the highest ceiling: it wins seed 44 and owns the single best run at 0.352 average EM. Over the training horizon, EM-only is also the least stable of the three: it alone carries a saw-tooth reward trajectory and a post-peak held-out slip, whereas both F1 shapes climb more smoothly (Section 4.3). A practitioner who wants reliable, low-variance behaviour across seeds should prefer F1+format; one chasing the best attainable endpoint should prefer F1-only.

The 0.1 floor does not mask the tool-use signal at 0.8B scale; it aids discovery. This inverts a pilot finding (the floor was expected to flatten the reward landscape and suppress tool use), and, at the seed we audited, the win decomposes as format-adherence plus retrieval-grounding rather than chain-coherence. The floor and the format-gate together push the policy toward emitting well-formed, retrieval-grounded answers, and that, not improved reasoning-chain consistency, is where the F1+format advantage comes from.

A higher training reward does not imply more exact matches, which is the substance of RQ6. F1+format posts the highest training reward yet the lowest exact-match rate, because the 0.1 floor inflates the scalar without lifting the underlying answer accuracy. Moreover, all three reward shapes share a training-time exact-match ceiling of roughly 20% to 22% (Results 4.3.6); this ceiling is a property of the model-and-data scale, not of the reward. The binding constraint is bridge-entity resolution under noisy retrieval, not reward design: none of the three reward shapes tested moves this 0.8B policy past the point where it can reliably chain intermediate entities through imperfect retrieved evidence.

The decisive recipe choices are few and stable across the ablation. The policy must be the hybrid checkpoint rather than the base model (the base model could not bootstrap the tool-call format in the 0.6B pilot); training on the single hardest dataset (MuSiQue) generalises to the held-out suite; a partial-credit reward with a format floor is preferable to sparse EM; and about one epoch suffices to clear the untrained floor decisively.

A further finding concerns emergent compression (RQ4); we state only its significance here, with the mechanism and numbers in Results 4.3. Under the unmodified outcome reward, the F1 shapes become leaner tool-callers (fewer searches and total tokens cut by roughly a quarter to a half) even as held-out EM rises. This is best explained by the 8192-token generation cap acting as an implicit length penalty: an emergent, not designed, mechanism that F1's dense partial credit exploits and EM-only's sparse 0/1 reward does not. The reduction is in the search and total-token budget, not in reasoning length (which follows the same early-dip-then-rise as the larger-scale runs); so at small scale the net rollout budget shrinks where Search-R1 and ReSearch report it growing (Chen et al., 2025), and EM-only is the exception that fails to compress.

Taken together, these results give small-LM practitioners working under realistic resource constraints a concrete, reproducible recipe; a clear negative result on the Search-R1-faithful EM-only reward; and the emergent-compression observation as a finding about how small models adapt under an unmodified outcome reward.

6.3 Cross-generation: post-training versus parameter count

RQ5 is a secondary, discussion-framed question: how does post-training a small, modern model compare against simply using a larger, older one? The reading from our results is that a current-generation sub-1B model plus reinforcement learning with verifi-

able rewards (RLVR) matches and slightly exceeds a roughly four times larger previous-generation 3B model on held-out multi-hop QA, evaluated on the same network. The trained Qwen3.5-0.8B endpoint (0.352 average EM) clears the untrained Qwen2.5-3B baseline (0.199) by +0.15 and edges our own Search-R1-style 3B GRPO reproduction (0.341) by +0.01, reached on roughly one twenty-sixth of the released 3B checkpoint's prompt exposure (one MuSiQue epoch versus about three epochs over an 8.5 times larger corpus). The modern-small plus RLVR bundle therefore outweighs a roughly fourfold parameter advantage here, and the far smaller training budget makes that reading more conservative, not less.

This stays a confounded observation, not an isolated claim. Model generation, architecture, the generation's own post-training, and our RLVR are bundled together, and the two endpoints used different RL recipes (our NeMo-RL (NVIDIA NeMo Team, 2025) reward-shape ablation versus Search-R1's verl (Sheng et al., 2024) EM-only run). The honest claim is therefore "a modern sub-1B model plus RLVR beats a four times larger previous-generation model", not "RLVR alone beats four times the parameters". Separating the generation's share from the parameter-count share would require a same-generation capability-frontier control, an untrained larger Qwen3.5 (say 2B) set against the RLVR-trained 0.8B, that we did not run; the within-model RLVR effect itself is already measured in Section 4.2 (the +0.260 untrained-to-trained 0.8B lift). Those controls, a 2B scale test, and the single-model, single-dataset, and three-seed caveats that bound every number above are discussed in Chapter 5.

6.4 Broader implications

Three implications reach beyond the specific runs reported here.

For RLVR, the result is that at sub-1B scale, with the training machinery held fixed, the reward *shape* alone is a decisive lever. The nine runs differ only in their reward, yet that single axis sets how well the recipe learns: it separates the best of the three shapes from the worst, even though all three clear the untrained floor by a wide margin. It does so under plain GRPO, without the algorithmic additions that dominate the large-model literature; a working recipe at this scale therefore does not require them. Whether those additions would add further gains on top is a lever we did not vary, and we leave it to future work. A sparse exact-match reward, the usual verifiable signal in the mathematics and code settings where RLVR was developed and the form Search-R1 carries into reason-over-search at 3B and above, does not transfer well downward: at this scale it is the dense partial-credit signal that best carries the learning gradient, whereas the sparse 0/1 reward gives a weak-prior policy a weaker, less informative signal and reaches a lower endpoint. In our runs this shows up directly as the sparse reward producing the weakest gradient updates of the three shapes at the matched training horizon (Chapter 4). The practical reading is that small-model RLVR needs its own reward-design study rather than a scaled-down copy of a large-model recipe.

For retrieval-augmented generation, the finding is that a sub-1B model can be trained to interleave reasoning with live search and to generalise across benchmarks from a single training corpus, with no distillation step from a larger teacher in the recipe. This lowers the entry cost of a reason-over-search agent to one modest training run. The emergent compression strengthens that reading: the better policies learn to search less and so run on fewer tokens overall, which directly reduces retrieval load and serving cost, so the accuracy gain does not come at the price of a more expensive agent.

For the broader study of small language models, the work adds a data point that the sub-1B regime is qualitatively its own. A reward-design finding can invert between our 0.6B pilot and the 0.8B main study (the 0.1 floor masked the tool-use signal in the pilot but aided discovery at 0.8B), and an efficiency effect can emerge in the opposite direction to the one the larger-scale literature reports. Conclusions drawn at 3B and above should not be assumed to hold below one billion parameters; the regime warrants direct study, which the single-model scope of this thesis only begins.

Bibliography

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2310.11511 (Oral).

Peter Belcak, Greg Heinrich, Shizhe Diao, Yonggan Fu, Xin Dong, Saurav Muralidharan, Yingyan Celine Lin, and Pavlo Molchanov. Small language models are the future of agentic ai, 2025. URL https://arxiv.org/abs/2506.02153.

Mingyang Chen, Linzhuang Sun, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z. Pan, Wen Zhang, Huajun Chen, Fan Yang, Zenan Zhou, and Weipeng Chen. Research: Learning to reason with search for LLMs via reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2503.19470.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, et al. Deepseek-r1: Incentivizing reasoning capability in LLMs via reinforcement learning. *Nature*, 645:633–638, 2025. doi: 10.1038/s41586-025-09422-z. arXiv:2501.12948.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The Faiss library, 2024. URL https://arxiv.org/abs/2401.08281.

Mehdi Fatemi, Banafsheh Rafiee, Mingjie Tang, and Kartik Talamadupula. Concise reasoning via reinforcement learning, 2025. URL https://arxiv.org/abs/2504.05185.

He, Qu, Liu, et al. JustRL: Scaling a 1.5B LLM with a simple RL recipe. ICLR 2026 Blogpost Track, 2025. URL https://arxiv.org/abs/2512.16649. arXiv:2512.16649.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING)*, 2020. arXiv:2011.01060 (2WikiMultiHopQA).

Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33 (1):117–128, 2011. doi: 10.1109/TPAMI.2010.57.

Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023. arXiv:2305.06983 (FLARE).

Bowen Jin, Jinsung Yoon, Priyanka Kargupta, Sercan O. Arik, and Jiawei Han. An empirical study on reinforcement learning for reasoning-search interleaved LLM agents, 2025a. URL https://arxiv.org/abs/2505.15117.

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. In *Conference on Language Modeling (COLM)*, 2025b. arXiv:2503.09516.

Jiajie Jin, Yutao Zhu, Xinyu Yang, Chenghao Zhang, and Zhicheng Dou. FlashRAG: A modular toolkit for efficient retrieval-augmented generation research. In *Companion Proceedings of the ACM Web Conference (WWW), Resource Track*, 2025c. doi: 10.1145/3701716.3715313. arXiv:2405.13576.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017. arXiv:1705.03551.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, 2020. URL https://aclanthology.org/2020.emnlp-main.550/.

Kotoge, Nishimura, and Ma. Can compact language models search like agents? distillation-guided policy optimization for preserving agentic rag capabilities. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2026. arXiv:2508.20324 (DGPO).

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics (TACL)*, 7:452–466, 2019.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, et al. Tulu 3: Pushing frontiers in open language model post-training. In *Conference on Language Modeling (COLM)*, 2025. arXiv:2411.15124.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. arXiv:2005.11401.

Gang Li, Yulei Qin, Xiaoyu Tan, Dingkang Yang, Yuchen Shi, Zihan Xu, Xiang Li, Xing Sun, and Ke Li. RoRecomp: Enhancing reasoning efficiency via rollout response recomposition in reinforcement learning, 2025. URL https://arxiv.org/abs/2509.25958. arXiv:2509.25958.

Zihan Liang, Yufei Ma, Ben Chen, Zhipeng Qian, Huangyu Dai, Lingtao Mao, Xuxin Zhang, Chenyi Lei, and Wenwu Ou. IG-Search: Step-level information gain rewards for search-augmented reasoning, 2026. URL https://arxiv.org/abs/2604.15148. arXiv:2604.15148.

Zichen Liu, Changyu Chen, Wenjun Li, Tianyu Pang, Chao Du, and Min Lin. Understanding r1-zero-like training: A critical perspective. In *Conference on Language Modeling (COLM)*, 2025. arXiv:2503.20783 (Dr. GRPO).

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023. arXiv:2212.10511 (PopQA).

NVIDIA NeMo Team. NeMo RL: A scalable and efficient post-training library. https://github.com/NVIDIA-NeMo/RL, 2025.

OpenAI. Learning to reason with LLMs. https://openai.com/index/learning-to-reason-with-llms/, 2024. Accessed: 2026-06-09.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. arXiv:2203.02155.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2023. arXiv:2210.03350 (Self-Ask; Bamboogle).

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Qwen Team. Qwen3.5: Towards native multimodal agents. Alibaba Qwen blog and Hugging Face model cards (Qwen/Qwen3.5-0.8B), 2026. URL https://qwen.ai/blog?id=qwen3.5.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. OpenAI technical report, 2019. URL https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2383–2392, 2016. doi: 10.18653/v1/D16-1264. URL https://aclanthology.org/D16-1264/.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019. arXiv:1908.10084.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2302.04761.

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, Mingchuan Zhang, Y. K. Li, Yu 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, 2024. URL https://arxiv.org/abs/2409.19256. verl; published at EuroSys 2025.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-searcher: Incentivizing the search capability in LLMs via reinforcement learning, 2025. URL https://arxiv.org/abs/2503.05592.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics (TACL)*, 10:539–554, 2022. arXiv:2108.00573.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023. arXiv:2212.10509 (IRCoT).

Wang, Asilis, Akgül, et al. Tina: Tiny reasoning models via LoRA, 2025a. URL https://arxiv.org/abs/2504.15777.

Wang, Yang, Zeng, et al. Reinforcement learning for reasoning in large language models with one training example, 2025b. URL https://arxiv.org/abs/2504.20571.

Fali Wang, Zhiwei Zhang, Xianren Zhang, Zongyu Wu, Tzuhao Mo, Qiuhao Lu, Wanjing Wang, Rui Li, Junjie Xu, Xianfeng Tang, Qi He, Yao Ma, Ming Huang, and Suhang Wang. A comprehensive survey of small language models in the era of large language models: Techniques, enhancements, applications, collaboration with llms, and trustworthiness, 2024. URL https://arxiv.org/abs/2411.03350.

Guoqing Wang, Sunhao Dai, Guangze Ye, Zeyu Gan, Wei Yao, Yong Deng, Xiaofeng Wu, and Zhenzhe Ying. Information gain-based policy optimization: A simple and effective approach for multi-turn search agents. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2510.14967 (IGPO).

Hongru Wang, Cheng Qian, Wanjun Zhong, Xiusi Chen, Jiahao Qiu, Shijue Huang, Bowen Jin, Mengdi Wang, Kam-Fai Wong, and Heng Ji. Acting less is reasoning more! teaching model to act efficiently, 2025c. URL https://arxiv.org/abs/2504.14870. arXiv:2504.14870 (OTC / OTC-PO).

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training, 2022. URL https://arxiv.org/abs/2212.03533. E5.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. arXiv:2201.11903.

Peilin Wu, Mian Zhang, Kun Wan, Wentian Zhao, Kaiyu He, Xinya Du, and Zhiyu Chen. HiPRAG: Hierarchical process rewards for efficient agentic retrieval augmented generation. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2510.07794.

Tianle Xia, Ming Xu, Lingxiang Hu, Yiding Sun, Wenwei Li, Linfang Shang, Liqun Liu, Peng Shu, Huan Yu, and Jie Jiang. Search-P1: Path-centric reward shaping for stable and efficient agentic RAG training. In *Proceedings of the Association for Computational Linguistics (ACL), Industry Track*, 2026. arXiv:2602.22576.

Yutao Xie, Nathaniel Thomas, Nicklas Hansen, Yang Fu, Li Erran Li, and Xiaolong Wang. TIPS: Turn-level information-potential reward shaping for search-augmented LLMs. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2603.22293.

Xu, Lu, Cheng, et al. How to train your deep research agent? prompt, reward, and policy optimization in Search-R1, 2026. URL https://arxiv.org/abs/2602.19526. arXiv:2602.19526.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018. arXiv:1809.09600.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023. arXiv:2210.03629.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, et al. DAPO: An open-source LLM reinforcement learning system at scale, 2025a. URL https://arxiv.org/abs/2503.14476.

Zhaochen Yu, Ling Yang, Jiaru Zou, Shuicheng Yan, and Mengdi Wang. Demystifying reinforcement learning in agentic reasoning, 2025b. URL https://arxiv.org/abs/2510.11701. arXiv:2510.11701.

Zhan, Li, Wang, et al. ExGRPO: Learning to reason from experience. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2510.02245.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. BERTScore: Evaluating text generation with BERT. In *International Conference on Learning Representations (ICLR)*, 2020. arXiv:1904.09675.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Junyang Lin, et al. Group sequence policy optimization, 2025. URL https://arxiv.org/abs/2507.18071.

A Full hyperparameter tables (training + eval)

The full GRPO (Shao et al., 2024) recipe and eval decode/retriever settings behind the main results.

- All nine runs (3 reward shapes x 3 seeds) share **one** recipe.
- The only axes that vary between runs are the **reward function** (F1-only / F1+format / EM-only; Appendix C) and the **seed** (42/43/44).
- Training is reported uniformly (mostly) as **4x RTX PRO 6000 Blackwell, DP=4**: the optimisation math is hardware-invariant, and the memory-side knobs (`train_micro_batch_size`, the TP-vs-DP topology) leave the GRPO math identical.

A.1 Training hyperparameters (GRPO recipe, all nine runs)

Table A.1: GRPO training recipe (per-knob value and rationale) shared by all nine runs of the Qwen3.5-0.8B search-tool ablation.

Knob	Value	Notes
Policy model	`Qwen/Qwen3.5-0.8B` (Qwen Team, 2026) (hybrid)	bf16; same checkpoint as the untrained floor
RL algorithm	GRPO	group-mean + std baseline (`use_leave_one_out_baseline: false`); matches ReSearch (Chen et al., 2025)
Group size G	5	`num_generations_per_prompt`
Prompts / step	64	paper uses 256; 64 x G=5 = **320 trajectories/step** (`train_global_batch_size: 320`). Divergence is memory, not algorithm
Train micro-batch	1	memory-only knob; compute identical to a larger micro-batch

Knob	Value	Notes
Max total sequence length	8192 tokens	prompt + response budget (`max_total_sequence_length`)
Max new tokens / turn	1024	per generation turn
Max rollout turns	10	**non-binding**; effectively length-bounded by the 8192-token budget (~5 to 7 turns typical)
KL coefficient beta	0.001	`reference_policy_kl_penalty`
KL estimator	k3	low-variance KL; verl (Sheng et al., 2024) `low_var_kl` equivalent
PPO (Schulman et al., 2017) clip epsilon	0.2 (symmetric)	`ratio_clip_min = ratio_clip_max = 0.2`
Optimizer	AdamW (fused)	
Learning rate	1e-6	constant, **no warmup**
Weight decay	0.01	
Rollout temperature / top_p	1.0 / 1.0	on-policy sampling for rollouts (eval is greedy; see Appendix A.2)
Reward	**F1-only / F1+format / EM-only**	the single ablated axis (Appendix C)
Seed	**42 / 43 / 44**	the second varied axis
Config epoch target	2 epochs (`max_num_steps: 622`)	**realized ~1 epoch**: every run stopped early by budget at step_312 (one full MuSiQue pass) or less; per-leg end-steps in Appendix D.2

Training-side retrieval (the rollout retriever): E5-base-v2 (Wang et al., 2022) over Wikipedia-2018, **IVF4096-SQ8** index, **top-5** (`top_n: 5`), per-chunk char safety cap `max_obs_chars: 1024` (the paper has no cap). IVF-SQ8 rather than flat because the flat index times out under concurrent rollout HTTP load; the full index tradeoff is in A.3 below.

A.2 Evaluation hyperparameters (held-out suite)

Every checkpoint of every run was evaluated identically. Decode is **greedy** (the convention of the published Search-R1 (Jin et al., 2025b) code, whose benchmark validation path disables sampling; ReSearch's (Chen et al., 2025) released evaluation configuration likewise sets `do_sample: false`; this is essential and matched).

Table A.2: Held-out evaluation hyperparameters: decode, retrieval, and context settings applied identically to every checkpoint of every run.

Knob	Value	Source
Benchmarks	NQ (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), PopQA (Mallen et al., 2023) (test); HotpotQA (Yang et al., 2018), 2Wiki (Ho et al., 2020), MuSiQue (Trivedi et al., 2022) (dev); Bamboogle (Press et al., 2023) (test)	51,713 rows / checkpoint
Decode	greedy (`temperature: 0.0`)	evaluation config
Max search turns	5	evaluation rollout loop
Per-turn generation budget	up to the 8192-token input budget (`step_limit`)	bounded by remaining context, not a fixed cap
Generator max input len	8192 tokens	matches SGLang `--context-length 8192`
Per-chunk observation cap	**120 tokens / chunk**	evaluation rollout loop; top-5 -> ~600 obs tokens/turn (per-chunk, not a total budget)
Stop tokens	`</tool_call>`, `</answer>`, `<|im_end|>`, `<|endoftext|>`	
`enable_thinking`	True	open `<think>` prefix each turn
Retrieval top-k	**5**	the evaluation entrypoint overrides the FlashRAG (Jin et al., 2025c) default of 3; the 3B Search-R1 reproduction (E.1) keeps top-3 for paper fidelity, which is the lone retrieval-depth confound in the cross-model comparison
Retrieval query max length	256 tokens	E5 query truncation
Retriever / corpus / index	E5-base-v2 / Wikipedia-2018 / IVF4096-SQ8	same retriever as training

The held-out eval renders through the **same** chat-template invocation and shares the answer/query parsers with the training rollout by re-export, so there is no train/eval drift to audit (the pipeline was built train/eval-aligned from the start; Appendix K).

A.3 Retriever and index choice (IVF-SQ8 vs flat)

The dense retriever (the E5-base-v2 embedder over the ~21M-passage Wikipedia-2018 corpus in the standard DPR (Karpukhin et al., 2020) 100-word chunking, served with FAISS (Douze et al., 2024)) can be configured two ways, and the choice is critical under training rollout load. We default to the **IVF4096-SQ8** quantised index for both training and eval; paper-fidelity eval can opt back into the exact **flat** index.

Table A.3: Retriever index comparison: resident RAM, recall@10, score fidelity, and per-query CPU latency for the exact flat index versus the default IVF4096-SQ8 index.

Index	Resident RAM	Recall@10	Scores	CPU latency / query
Flat (exact inner-product)	65 GB	100 %	exact	100 to 300 ms
IVF4096-SQ8 (default)	**~16 GB**	**96 to 99 %**	very close	**30 to 100 ms** (3 to 10x faster)

All numbers at top-10, `nprobe=64`. The corpus itself is a separate ~14 GB memory-mapped, page-cache-evictable cost; the index is what dominates resident memory.

How these numbers were obtained. The two RAM figures are exact arithmetic on the corpus, not paper values: 21,015,324 passages x 768-dim E5 vectors at 4 bytes (fp32, flat) = 64.6 GB, and at 1 byte (8-bit scalar quantisation) = 16.1 GB, matching the 65 GB / ~16 GB above. The recall@10 range and the per-query latencies are engineering figures observed during index bring-up on the evaluation workstation (FAISS retrieval runs on its CPU), not a logged, reproducible experiment, and we label them accordingly. What is citable is the methodology behind them: the inverted-file (IVF) structure with scalar / product quantisation (Jégou et al., 2011) as implemented in FAISS, over E5 embeddings of the DPR Wikipedia-2018 corpus.

Three reasons this is the right default, and one honest caveat:

1. **Concurrency, not single-query speed, is the binding constraint.** The GRPO rollout loop fires many `/search` calls concurrently; the flat index (65 GB, 100 to 300 ms/query on CPU) times out under that load, whereas IVF-SQ8 scans only `nprobe=64` of 4096 cells and stays reliably sub-100 ms. This is why training cannot simply use the exact index.
2. **Memory: ~75 % smaller.** SQ8 scalar-quantises each vector from fp32 to 8 bits, cutting the resident index from 65 GB to ~16 GB, which is what lets the retriever co-reside with the rest of the pipeline on one host.
3. **Downstream risk is bounded and common-mode.** Recall@10 gives up a few points against flat (the 96 to 99 % engineering range above). We did not run a controlled flat-versus-IVF end-to-end evaluation, so we make no indistinguishability claim; two structural arguments bound the residual risk instead: E5 produces peaky similarity distributions (the top candidates are clearly separated, so SQ8 mostly

reshuffles among already-strong matches), and an agent that can issue follow-up searches absorbs a rare miss. Decisively for the headline result, every reward-shape comparison (all nine runs, every checkpoint, and the untrained floor) is evaluated through the same IVF-SQ8 index, so any recall loss is common-mode: it cannot reorder the reward shapes.

Why `nlist=4096` specifically. FAISS's `min_points_per_centroid=39` floor needs >= 2.55M training rows for `nlist=65536` but only ~160K for `nlist=4096`; with our 1M-row training sample, 4096 cells sit comfortably above the floor, whereas 65536 would have trained on a thin sample with degraded centroids. `nprobe=64` is the SQ8 accuracy-vs-latency sweet spot.

Caveat (fairness). Both source papers (Search-R1, ReSearch) use exact retrieval. This IVF-vs-flat split is ours, so it slightly favours the papers in any head-to-head retrieval-quality comparison; it is a deliberate engineering tradeoff for training throughput, not a quality choice.

A fuller index-family comparison (HNSW, IVF-PQ variants, GPU latencies) is omitted here; the flat and IVF4096-SQ8 indexes above are the only two used in this study.

A.4 The training-software overlay (component inventory)

The Method overview (Method 3.3) names the task-level pieces; this is the complete inventory. Stock NeMo-RL provides the GRPO trainer, the rollout loop, and generic environment interfaces, with bundled example environments for mathematics, code execution, and games; it ships no Search-R1-style retrieval task, no HTTP retriever client, and no QA reward. The overlay adds, as self-contained modules registered with the framework's component registries:

- a **dataset adapter** that wraps the Search-R1-format MuSiQue parquet into the framework's dataset interface and tags each row with its task route;
- the **multi-turn search environment**: parses the rollout's latest tool call, issues a batched HTTP query to the retriever service, formats the returned chunks as `<tool_response>` turns, and terminates the episode on a parsed `<answer>` or at the turn cap;
- the **reward modules** for the three shapes (Appendix C), re-exported from the evaluation scorer;
- the **prompt/template processor** that renders each data row through `apply_chat_template` with the `search` tool schema (Appendix B);
- the **registry wiring** that registers the dataset, environment, processor, and reward with NeMo-RL's component registries at import time.

Three small patches to the vendored framework round this out: extra rollout metrics (per-step tool-call counts and a valid-answer signal), a lengthened weight-refit timeout for the DP=4 topology, and a guard in the sharded-checkpoint offload path. None of the three alters the GRPO maths.

A.5 GRPO vs PPO optimiser memory (why no value critic)

GRPO's group-relative baseline replaces PPO's learned value critic, so it trains one network instead of two. The saving, at our 0.8B policy in bf16 with AdamW (fp32 moments):

Table A.4: Optimiser-memory accounting for the 0.8B policy: PPO with a value critic versus GRPO with a group-relative baseline and no critic.

Component	PPO	GRPO (ours)
Policy weights (bf16, ~2 B/param)	~1.6 GB	~1.6 GB
Policy AdamW moments m+v (fp32, ~8 B/param)	~6.4 GB	~6.4 GB
Value critic weights (second policy-sized net)	~1.6 GB	**none**
Value critic AdamW moments m+v	~6.4 GB	**none**
Extra critic forward/backward per step + activations	yes	**none**

So a same-sized PPO critic adds on the order of ~8 GB of optimiser/critic state (plus its own activations and a second forward/backward) that GRPO never allocates (the PPO column is the conventional same-size-critic assumption, not a measured baseline); on a single modest GPU this is often the difference between fitting the run and not. GRPO's only added cost is sampling G=5 rollouts per prompt to form the group baseline (Method 3.3).

B Prompt templates and tool-call format

The same prompt drives the training rollout and the held-out eval (byte-aligned by construction; A.2). There is **no hand-authored system file**; the user prompt is terse (170 chars), and `apply_chat_template(..., tools=[search], enable_thinking=True)` auto-injects the tool schema as a system block (~220 words of fixed prefix per rollout). That block is **in-distribution** for Qwen3.5's post-training, which is the reason we use the family-native nested-XML tool-call format rather than the papers' invented string tags (`<search>`/`<information>` in Search-R1, `<search>`/`<result>` in ReSearch; both are off-distribution for Qwen3.5, and the 0.6B pilot showed they are free only at matched step count, H.4). The block below is **byte-exact**, rendered from the `Qwen/Qwen3.5-0.8B` tokenizer with the `search` tool object and the terse prompt:

```
<|im_start|>system
# Tools

You have access to the following functions:

<tools>
{"type": "function", "function": {"name": "search", "description":
"Search Wikipedia for passages relevant to the query. Returns the top-K
most relevant chunks. Call this whenever the question requires factual
knowledge you do not already have.", "parameters": {"type": "object",
"properties": {"query": {"type": "string", "description": "The search
query. Be specific."}}, "required": ["query"]}}}
</tools>

If you choose to call a function ONLY reply in the following format
with NO suffix:

<tool_call>
<function=example_function_name>
<parameter=example_parameter_1>
value_1
</parameter>
<parameter=example_parameter_2>
This is the value for the second parameter
that can span
multiple lines
</parameter>
</function>
</tool_call>

<IMPORTANT>
Reminder:
- Function calls MUST follow the specified format: an inner
```

```
<function=...></function> block must be nested within
<tool_call></tool_call> XML tags
- Required parameters MUST be specified
- You may provide optional reasoning for your function call in natural
language BEFORE the function call, but NOT after
- If there is no function call available, answer the question like
normal with your current knowledge and do not tell the user about
function calls
</IMPORTANT><|im_end|>
<|im_start|>user
Use the ‘search‘ tool to look up facts as needed. When you have the
answer, write it inside <answer> and </answer>. For example,
<answer> Beijing </answer>.
Question: {question}<|im_end|>
<|im_start|>assistant
<think>
```

Tag scheme. The policy emits a search as a `<tool_call>` wrapping an inner `<function=search>` with a `<parameter=query>...</parameter>` body (closed by `</function></tool_call>`); each retrieved chunk returns in its own `<tool_response>...</tool_response>` block, all stacked inside one turn-bounded `<|im_start|>user ... <|im_end|>` message, after which generation resumes from `<|im_start|>assistant`. The final answer is plain `<answer>X</answer>` (no `\boxed{}`; the scorer accepts both). `<tool_call>` / `<tool_response>` are native Qwen3.5 vocabulary tokens, not invented strings.

C Reward design and construction

The reward is the single ablated axis (Method 3.4; Appendix A.1). All three rewards score the **last** `<answer>...</answer>` block of the rollout, after SQuAD-style normalisation, and share the scorer with the held-out eval by **re-export** (the training reward and the eval metric call the same `normalize_answer` / `em_check` / `f1_check`), so train/eval scoring cannot drift by construction.

Shared primitives. `normalize_answer` is the SQuAD (Rajpurkar et al., 2016) v1.1 answer normalisation: lowercase, strip punctuation, remove the articles a/an/the, collapse whitespace. On the normalised strings: - **EM** (`em_check`): 1.0 if the normalised prediction equals any normalised gold answer, else 0.0. - **token-F1** (`f1_check`): SQuAD-style token-overlap F1. Tokens are the whitespace split of the normalised string; precision = |pred ∩ gold| / |pred|, recall = |pred ∩ gold| / |gold|, F1 = 2PR/(P+R), best over a multi-answer gold list (both-empty = 1.0, one-empty = 0.0). No stop-word strip, matching the Search-R1 / ReSearch convention.

```
extract_solution(rollout): content of the LAST <answer>...</answer> block, or None
```

F1-only (ReSearch's shape minus the floor; the dense intermediate).

```
answer = extract_solution(rollout)
reward = 0.0                    if answer is None      # no parseable answer
       = f1_check(answer, gold) otherwise              # 0.0 .. 1.0, dense
```

F1 + 0.1 format (ReSearch's 3-tier shape, non-additive). The 0.1 format floor *replaces* F1 when F1=0; it is not an additive bonus (F1=0.5 gives reward 0.5, same as F1-only):

```
if not is_valid_format(rollout):    reward = 0.0            # format broken
elif extract_solution(rollout) is None:  reward = 0.1       # format ok, no <answer>
elif f1(answer, gold) == 0:         reward = 0.1            # format ok, answer
wrong
else:                               reward = f1(answer, gold)
```

`is_valid_format` ports ReSearch's `validate_template_format` to our schema; it requires: (1) `<think>` closed and balanced; (2) `<tool_call>` tags balanced; (3) each tool-call body is a valid `<function=NAME>...<parameter=...>...</function>` block; (4) `<answer>` open-count == close-count with >= 1 closed block; (5) the rollout, stripped, **terminates on `</answer>`**.

How our F1+format differs from ReSearch (reward-shape-identical, not byte-identical). ReSearch's format gate is the same idea ported to a different surface syntax:

Table C.1: How our F1+format reward differs from ReSearch's: the same reward-shape components mapped onto our tool-call surface syntax.

	ReSearch	F1+format (ours)
Answer scaffold	`\boxed{X}` inside `<answer>`	plain `<answer>X</answer>`, no `\boxed{}`
Terminal gate	response ends with EOS + contains `\boxed`	response ends with `</answer>` (the vLLM stop string)
Answer extraction	inside `\boxed{...}`	inside the LAST `<answer>...</answer>`
Tool-call body validated	JSON	Qwen3.5 nested-XML `<function=...>` `<parameter=...>`
3-tier values	0 / 0.1 / F1	0 / 0.1 / F1 (identical)

The two disagree only on rollouts that are otherwise valid but contain no `\boxed{}`; the shaping *values* are identical. Dropping `\boxed{}` is a deliberate train/eval-parity choice (the scorer accepts both shapes).

EM-only (Search-R1's paper-faithful reward), routed through the shared training-reward module with all shaping coefficients pinned to 0:

```
answer = extract_solution(rollout)
reward = 1.0   if answer is not None and em_check(answer, gold) == 1
       = 0.0   otherwise        # no <answer>, format broken, or wrong answer
```

This reproduces Search-R1's `r(x, y) = EM(a_pred, a_gold)`. It is **value-equivalent, not byte-identical**: Search-R1's `extract_solution` returns None when `len(matches)` $\leq$ `1` (to discard a prompt-embedded worked-example `<answer>`), whereas ours uses $\leq$ `0` because our `solution_str` already excludes the user prompt; the two agree on every single-answer rollout but would differ on a prompt that re-embeds an example answer.

The third metric reported in Results 4.1, **accuracy (substring-cover / SubExactMatch)**, is an *eval* metric, not a reward: 1.0 if a normalised gold answer is a substring of the normalised prediction, else 0.0 (more lenient than EM). It is computed and published for all checkpoints but is never optimised against.

D Held-out eval detail (backs Results 4.1)

The full held-out detail behind Results 4.1.

D.1 Per-benchmark EM at each run's best checkpoint

Best checkpoint of each run within its seed horizon (42 $\leq$ 180, 43 $\leq$ 230, 44 $\leq$ 312; step in the avg-EM column), broken out over the 7 benchmarks, with avg F1 and avg ACC at that checkpoint. This is the per-benchmark detail summarised by the body 3x3 tables.

Table D.1: Per-benchmark EM at each run's seed-horizon best checkpoint, with average F1 and ACC at that checkpoint, for all nine reward-shape by seed runs.

Run	NQ	TriviaQA	PopQA	HotpotQA	2Wiki	MuSiQue (train)	Bamboogle	avg EM @step	avg F1	avg ACC
F1-only s42	0.301	0.492	0.319	0.309	0.287	0.130	0.352	0.313 @110	0.402	0.394
F1-only s43	0.301	0.489	0.321	0.259	0.274	0.093	0.296	0.291 @210	0.383	0.355
F1-only s44	0.363	0.531	0.370	0.359	0.285	0.161	0.392	**0.352** @310	0.434	0.397
F1+format s42	0.319	0.500	0.335	0.311	0.313	0.117	0.360	0.322 @100	0.413	0.378
F1+format s43	0.335	0.533	0.352	0.322	0.296	0.092	0.264	0.313 @210	0.400	0.359
F1+format s44	0.310	0.521	0.340	0.307	0.278	0.121	0.288	0.309 @270	0.394	0.355
EM-only s42	0.314	0.492	0.342	0.306	0.238	0.103	0.312	0.301 @170	0.368	0.334
EM-only s43	0.289	0.479	0.321	0.243	0.236	0.071	0.208	0.264 @120	0.333	0.293
EM-only s44	0.343	0.524	0.383	0.315	0.283	0.110	0.264	0.318 @310	0.386	0.340

Per-benchmark reads: TriviaQA highest everywhere (~0.48-0.53), MuSiQue lowest (~0.07-0.16, the hardest 3-hop set and the only in-distribution dataset, since all nine runs train on MuSiQue only); the other six benchmarks are out-of-distribution, so the held-out generalisation claim rests on them. PopQA is where EM-only is least disadvantaged (short-string entity answers reward exact match). On the held-out (OOD) datasets, F1+format wins the EM-metric count (5 of 7 datasets at seed 42); EM-only wins only PopQA.

D.2 Exact per-leg end-steps and the matched-horizon definition

The body says "trained to roughly 180 / 230 / 312"; the exact final checkpoint step per leg:

Table D.2: Exact final-checkpoint step per leg (reward shape by seed); the basis for the matched-horizon definition.

Seed	F1-only	F1+format	EM-only
42	180	180	210
43	230	250	240
44	310	310	310

(Training reached ~312 on seed 44; checkpoints are saved every 10 steps to step_310.) **Standard horizons** cap each seed at 180 / 230 / 312; the three legs that ran past their seed cap are truncated to it: F1+format seed-43 reached avg EM 0.317 at step_250 but is reported at its ≤230 best (0.313 @ step_210), EM-only seed-43 ran to step_240 (best at step_120, so unaffected), and EM-only seed-42 ran to step_210 but its best is at step_170 (so unaffected). "Matched horizon" = best checkpoint at step ≤ 180, the largest horizon common to all nine legs; used for every cross-seed claim. "Seed horizon" = best within each seed's cap.

D.3 avg F1 and ACC per run

avg F1 at the best-F1 checkpoint (matched ≤ 180 / seed horizon):

Table D.3: Average token-F1 at each run's best-F1 checkpoint (matched horizon / seed horizon), per reward shape and seed.

Reward shape	seed 42	seed 43	seed 44
F1-only	0.402 / 0.402	0.369 / 0.383	0.407 / 0.434
F1+format	0.413 / 0.413	0.370 / 0.400	0.390 / 0.394
EM-only	0.368 / 0.368	0.333 / 0.333	0.319 / 0.386

avg ACC (substring-cover) at each run's seed-horizon best-EM checkpoint:

Table D.4: Average substring-cover ACC at each run's seed-horizon best-EM checkpoint, per reward shape and seed.

Reward shape	seed 42	seed 43	seed 44	mean
F1-only	0.394	0.355	0.397	0.382
F1+format	0.378	0.359	0.355	0.364

Reward shape	seed 42	seed 43	seed 44	mean
EM-only	0.334	0.293	0.340	0.323

EM-only is worst on F1 and ACC at every seed, with a larger deficit than on EM (it does not optimise partial overlap). The metric ordering (EM / F1 / ACC) agrees throughout.

D.4 Per-dataset best-checkpoint spread

At the seed horizon, for every run the best-average-EM checkpoint equals the best-average-F1 checkpoint (no EM/F1 trade-off), but no single checkpoint is best on all 7 datasets: only 1-6 of 7 datasets peak at the best-average checkpoint, and per-dataset peaks scatter. (This is a within-run checkpoint-selection note, not a main-text result.)

Table D.5: Per-dataset best-checkpoint spread: for each run, the best-average checkpoint and how many of the seven datasets peak there.

Run	best-avg ckpt	#datasets peaking there	per-dataset peak-step range
F1-only s42	step_110	4/7	100-180
F1+format s42	step_100	1/7	100-180
EM-only s42	step_170	3/7	50-180
F1-only s43	step_210	1/7	90-210
F1+format s43	step_210	3/7	190-230
EM-only s43	step_120	3/7	120-170
F1-only s44	step_310	4/7	220-310
F1+format s44	step_270	3/7	110-270
EM-only s44	step_310	6/7	70-310

(Full-epoch seed-44 runs concentrate near the end only because they were still climbing.)

D.5 Full per-checkpoint sweeps

The complete per-checkpoint x per-dataset EM/F1(/ACC) tables are the published evaluation artifact (Appendix K): the raw per-cell eval outputs (metric score + config) are public on HF (`pantomiman/reason-over-search-eval-m5`; the full artifact list is in Appendix K), together with the aggregated per-run CSVs.

D.6 F1 and ACC trajectory figures + consolidated three-metric table

The §4.1 body shows the held-out **EM** figures (Figures 4.1 to 4.5) plus the cross-metric robustness bar (Figure 4.6), which is the cross-seed mean of the matched-horizon consoli-

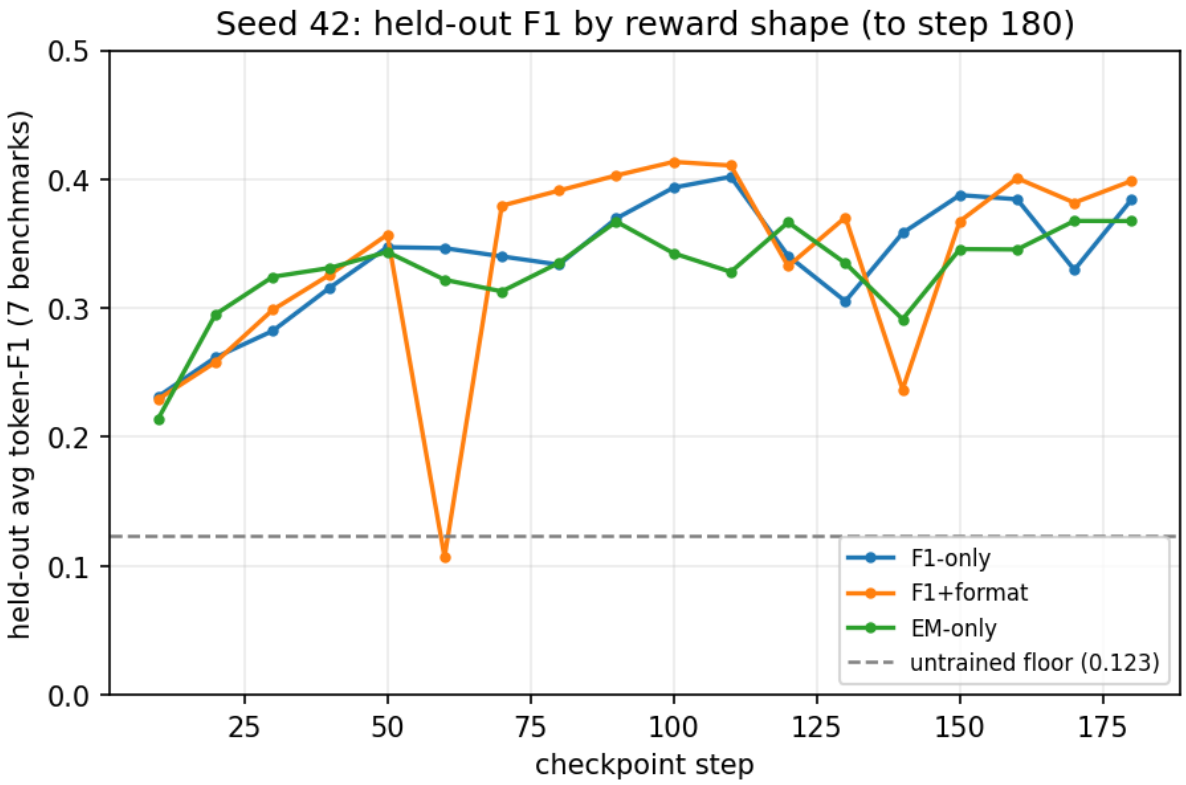


Figure D.1: Seed 42 token-F1 (to step 180).

dated table at the end of this section. The same five figures for **token-F1** and **substring-cover ACC** are below, regenerated by the same plotting script; each metric uses its own untrained terse floor as the dashed reference (**EM 0.092, F1 0.123, ACC 0.146**; Results 4.1). The reward-shape ordering is the same on all three metrics (EM-only worst at matched horizon; F1+format the most seed-robust; F1-only the highest ceiling), which is the §4.1.4 "not an EM artifact" check. **What the figures show:** the F1 and ACC trajectories track the EM trajectories point-for-point, including the same features (the seed-43 F1-only collapse over steps ~130-190, the per-cadence oscillations, the EM-only post-peak decay) on every metric; the dense metrics sit a fixed offset above EM (F1 highest, then ACC, then EM) because partial-credit / substring matches are strictly more lenient. No metric reorders the reward shapes, so the §4 conclusions are metric-independent.

Token-F1. Figures D.1 to D.5 repeat the five held-out views on token-F1.

Substring-cover ACC. Figures D.6 to D.10 repeat the same five views on substring-cover ACC.

Consolidated three-metric table. EM, F1, and ACC together at each run's **best-avg-EM checkpoint** (one checkpoint, three metrics, so the metrics are directly comparable rather than each reported at its own best step).

Matched horizon (best avg-EM ckpt at step $\leq$ 180):

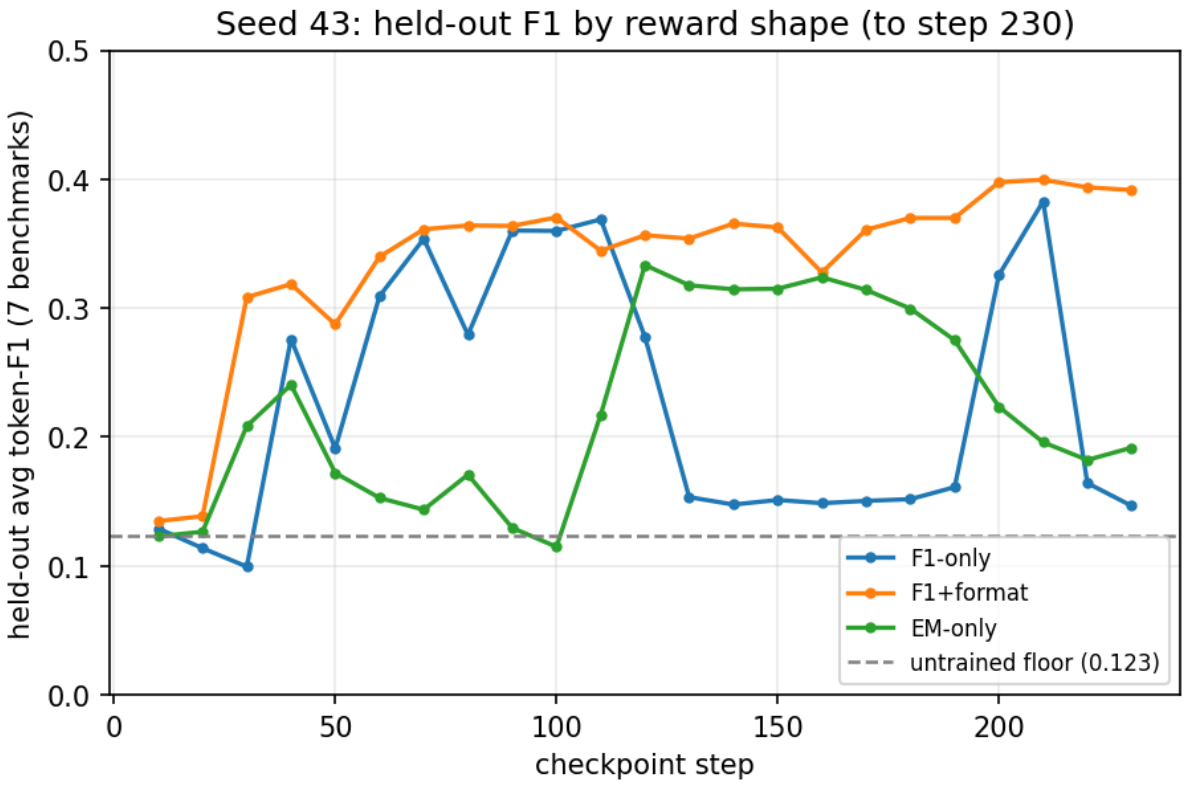


Figure D.2: Seed 43 token-F1 (to step 230).

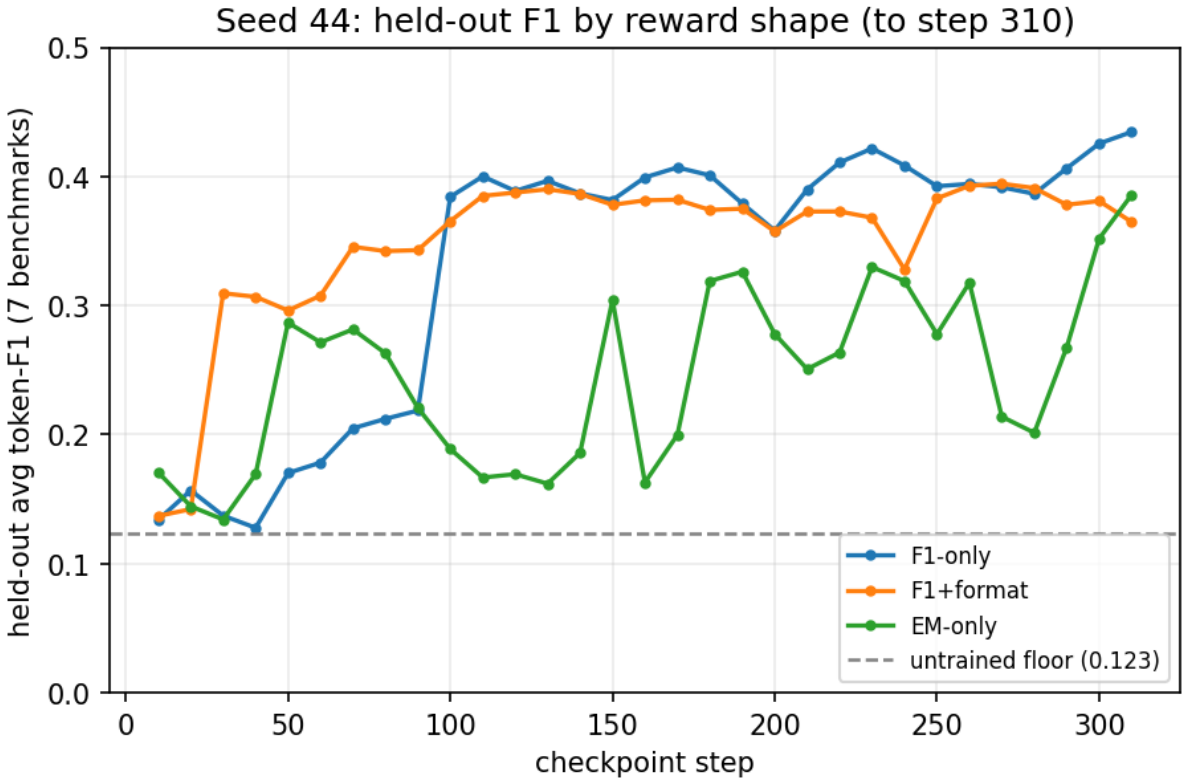


Figure D.3: Seed 44 token-F1 (full epoch, to step 310).

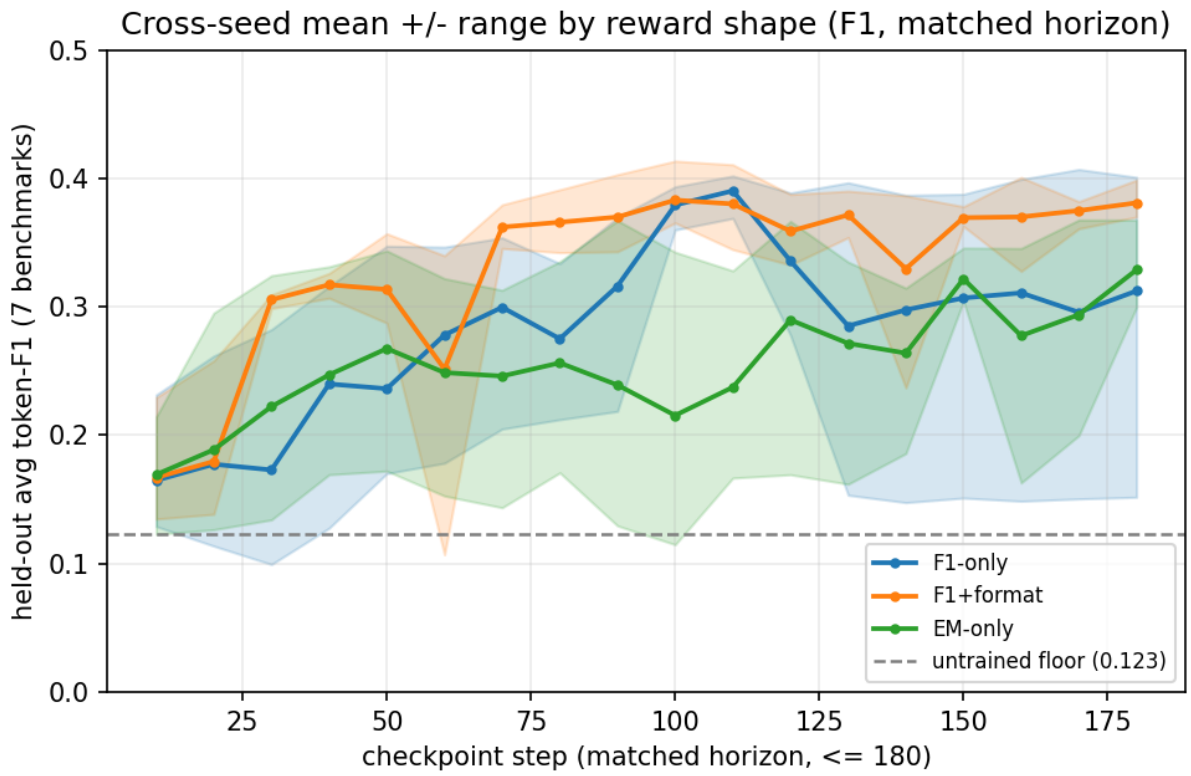


Figure D.4: Cross-seed mean ± range, token-F1, matched to step ≤ 180.

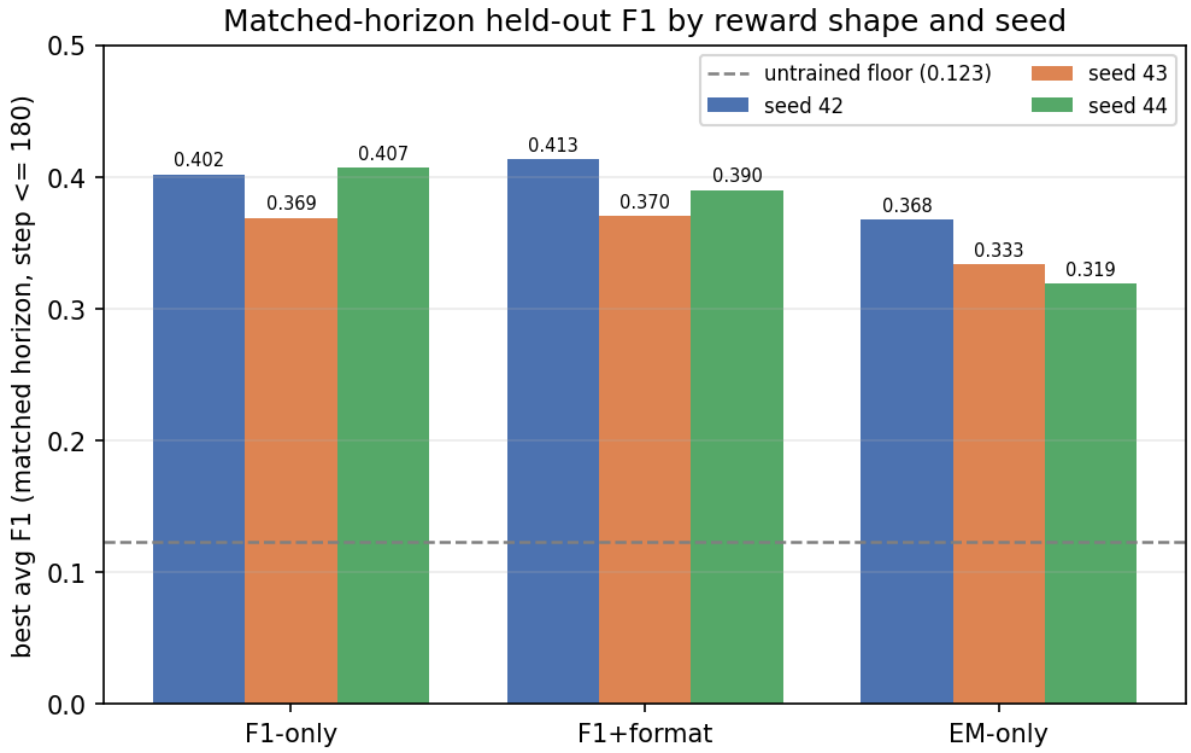


Figure D.5: Matched-horizon best token-F1, grouped by reward shape, one bar per seed.

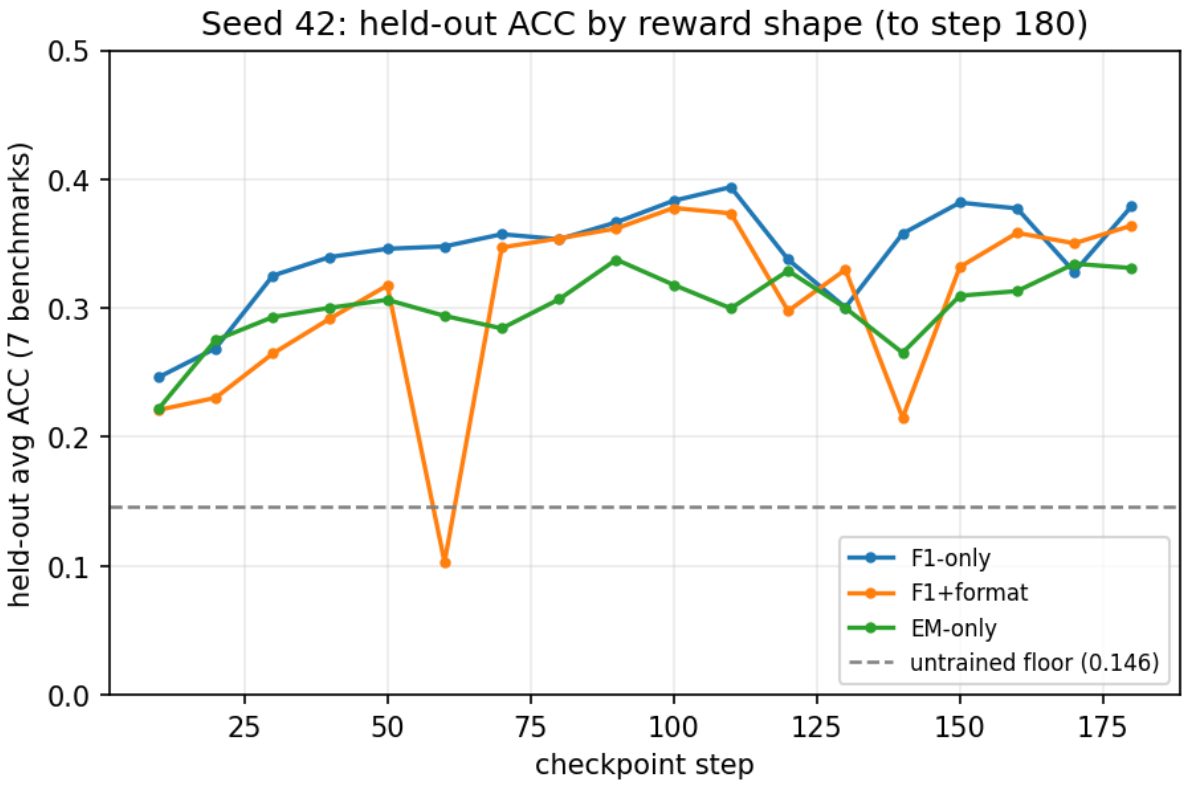


Figure D.6: Seed 42 ACC (to step 180).

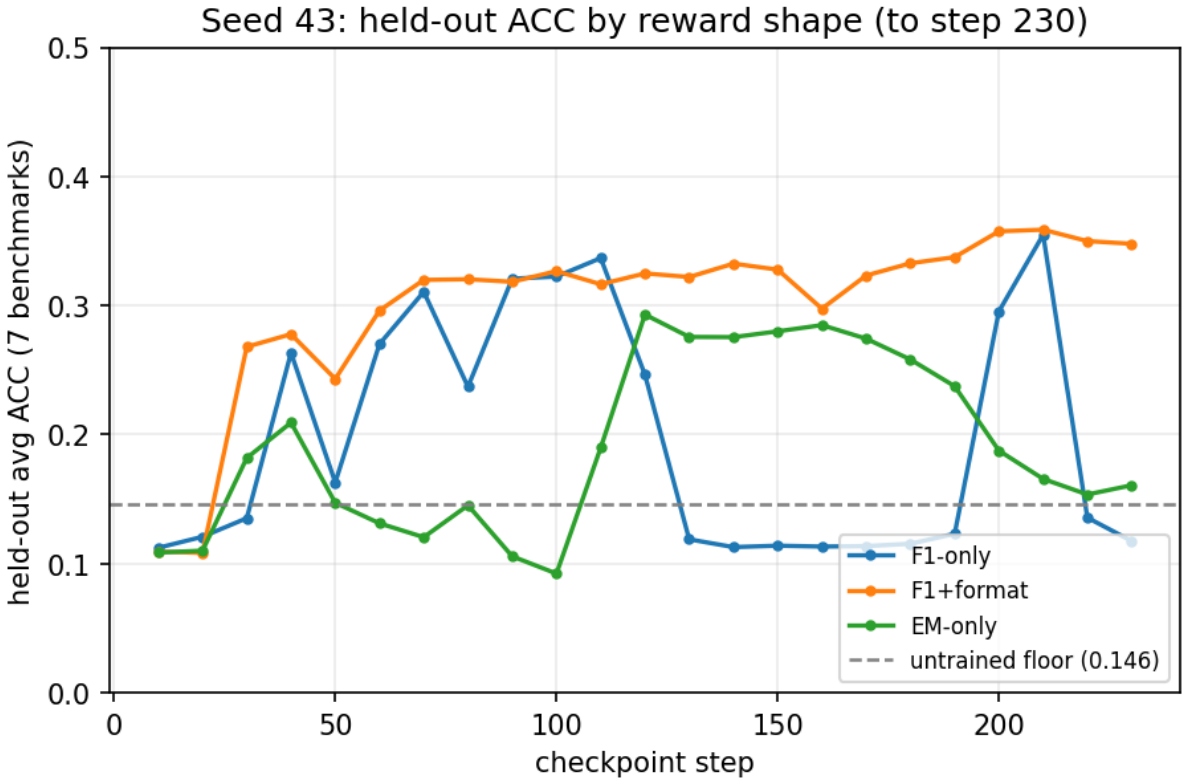


Figure D.7: Seed 43 ACC (to step 230).

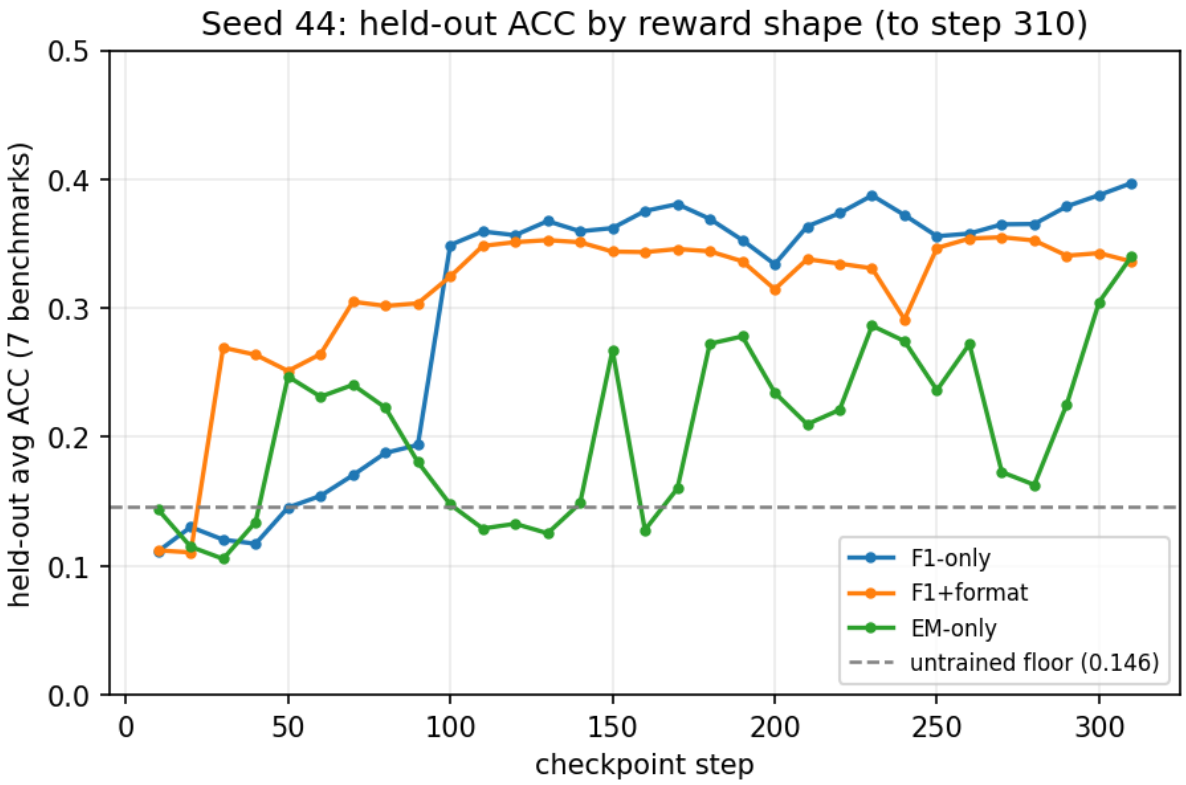


Figure D.8: Seed 44 ACC (full epoch, to step 310).

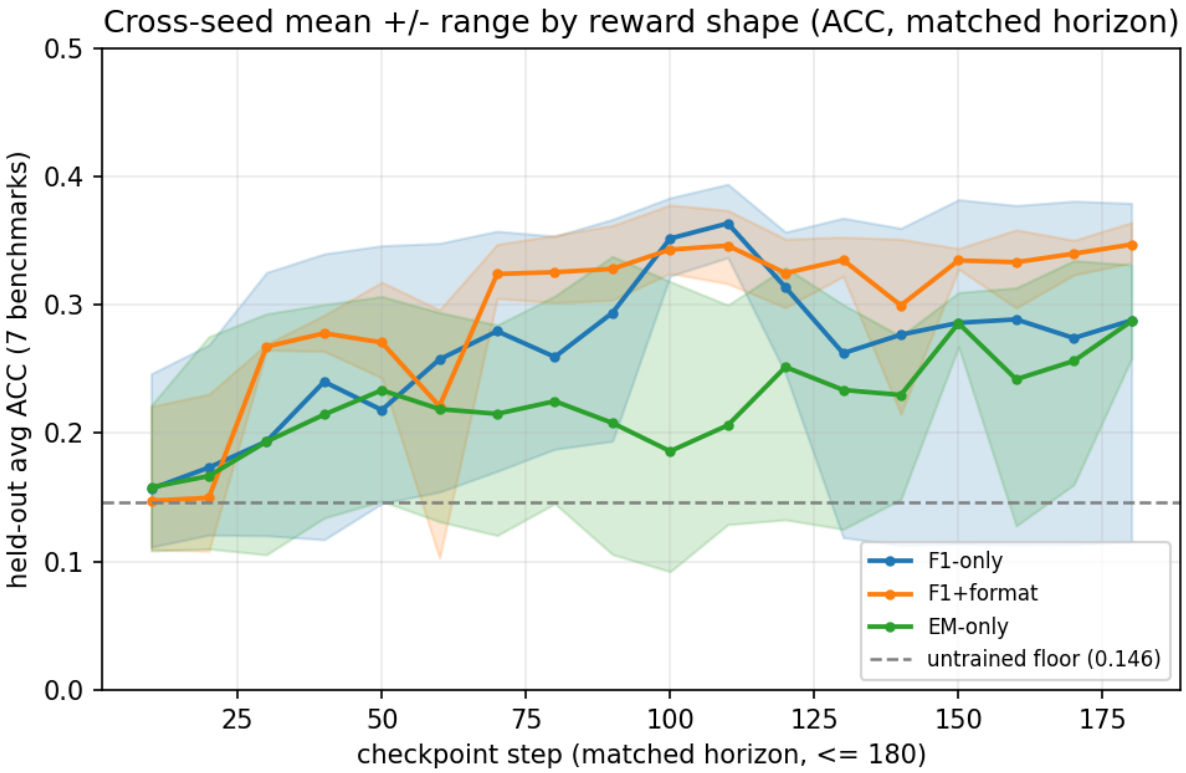


Figure D.9: Cross-seed mean ± range, ACC, matched to step ≤ 180.

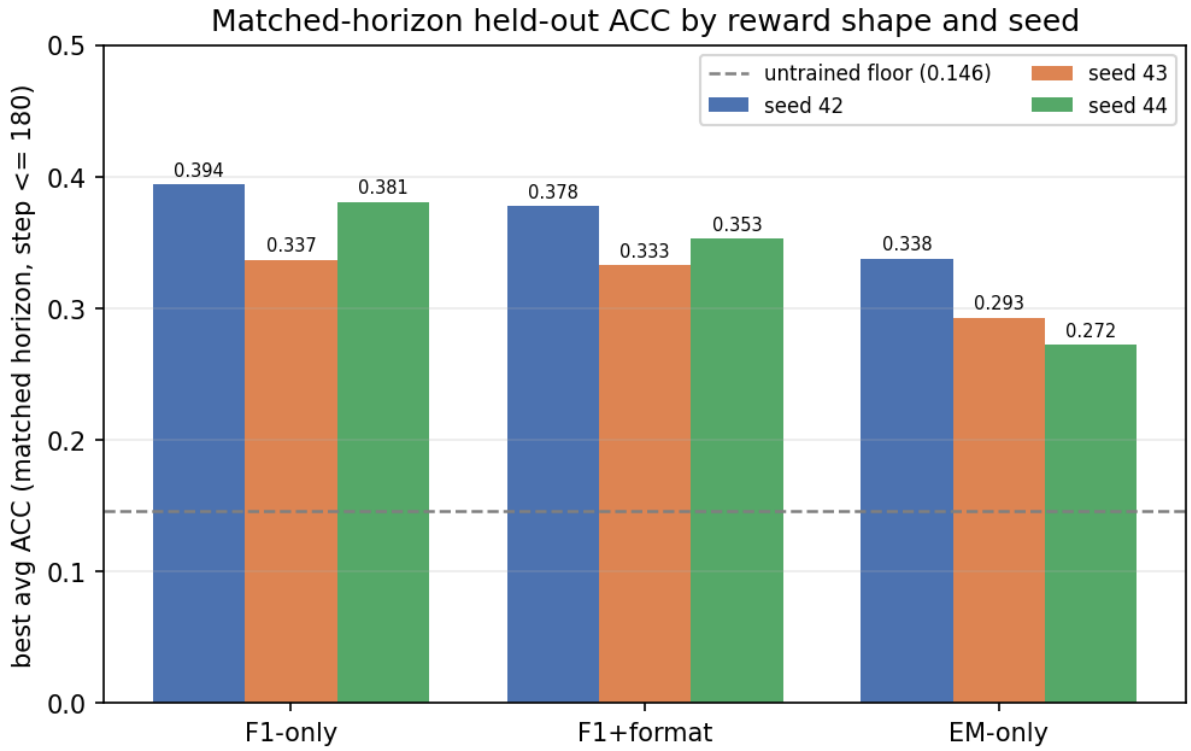


Figure D.10: Matched-horizon best ACC, grouped by reward shape, one bar per seed.

Table D.6: Consolidated EM, F1, and ACC at each run's matched-horizon (step $\leq$ 180) best-average-EM checkpoint.

Reward shape	seed	step	EM	F1	ACC
F1-only	42	110	0.313	0.402	0.394
F1-only	43	110	0.283	0.369	0.337
F1-only	44	170	0.318	0.407	0.381
F1+format	42	100	0.322	0.413	0.378
F1+format	43	180	0.292	0.370	0.333
F1+format	44	110	0.308	0.385	0.348
EM-only	42	170	0.301	0.368	0.334
EM-only	43	120	0.264	0.333	0.293
EM-only	44	180	0.249	0.319	0.272

Seed horizon (best avg-EM ckpt within each seed's cap, 42 $\leq$ 180 / 43 $\leq$ 230 / 44 $\leq$ 312):

Table D.7: Consolidated EM, F1, and ACC at each run's seed-horizon best-average-EM checkpoint.

Reward shape	seed	step	EM	F1	ACC
F1-only	42	110	0.313	0.402	0.394
F1-only	43	210	0.291	0.383	0.355

Reward shape	seed	step	EM	F1	ACC
F1-only	44	310	**0.352**	**0.434**	**0.397**
F1+format	42	100	0.322	0.413	0.378
F1+format	43	210	0.313	0.400	0.359
F1+format	44	270	0.309	0.394	0.355
EM-only	42	170	0.301	0.368	0.334
EM-only	43	120	0.264	0.333	0.293
EM-only	44	310	0.318	0.386	0.340

On the matched horizon EM-only is lowest on all three metrics at every seed. On the seed horizon the one exception is seed-44 EM (0.318, edging F1+format's 0.309, the length effect discussed in §4.1); its F1 and ACC stay lowest there too. The EM / F1 / ACC columns otherwise move together (no metric inversion), so the §4.1 EM-based conclusions hold on the dense (F1) and lenient (ACC) metrics too.

D.7 Statistical read of the three-seed ranking

The headline ranking (EM-only last at all three seeds at the matched horizon) admits an exact test despite the small number of seeds. Under the null hypothesis that reward shape has no effect on held-out average EM, the three shapes are exchangeable within each seed: each shape's rank is uniform over the three positions, independently across the three seed replicates. The probability that a pre-specified shape ranks last at all three seeds is then $(1/3)^3 = 1/27 \approx 0.037$, significant at the 5% level. EM-only is pre-specified in the required sense: it is the Search-R1-faithful reward whose transfer the ablation was designed to test, an arm fixed before any held-out number was read, not selected after seeing the results (the pilot in fact predicted the opposite direction, so there was no fishing toward this outcome). Two disclosures. First, the conservative variant that does not single out a shape in advance (the probability that *some* shape ranks last at all three seeds) is $3 \times 1/27 = 1/9 \approx 0.111$; we consider the pre-specified read the correct one, but report both. Second, the same last-place ranking holds on F1 and ACC at every seed (the consolidated table above), which corroborates the result but is not independent evidence (the three metrics score the same predictions), so we do not combine p-values across metrics.

The test has no power for the secondary ordering. Between the two F1 shapes, a 2-to-1 seed split (F1+format wins seeds 42 and 43, F1-only wins seed 44) is uninformative at $n = 3$: even a 3-to-0 sweep would only reach a one-sided sign-test $p = (1/2)^3 = 0.125$. This is why §4.1 reports the F1-only versus F1+format ordering as seed-dependent rather than as a verdict.

D.8 Held-out search-count profile and the termination gap

At each run's best-EM checkpoint we recovered, for every held-out question on the four multi-hop benchmarks (MuSiQue, 2WikiMultiHopQA, HotpotQA, Bamboogle), the number of search calls the policy issued and whether its answer was an exact match; pooling the three seeds gives 67,569 graded rollouts per reward shape. Two patterns emerge.

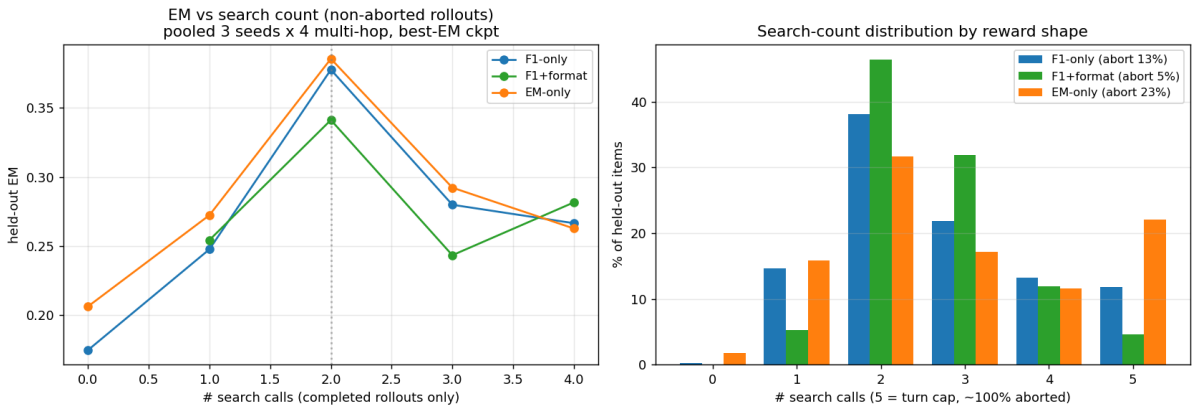


Figure D.11: Held-out behaviour by search count at each run's best-EM checkpoint, pooled over three seeds and the four multi-hop benchmarks (67,569 rollouts per reward shape). Left: mean exact-match against the number of search calls for completed (non-aborted) rollouts; all three reward shapes peak at two searches. Right: the distribution of search counts per reward shape with the per-shape abort rate; EM-only places far more mass in the five-search turn-budget bin (about 100% aborted) than the F1 shapes, the main source of its lower held-out exact-match.

First, held-out exact-match as a function of search count is an inverted-U that peaks at two searches for all three reward shapes and in every one of the four datasets (Figure D.11, left). Among rollouts that terminate with a parseable answer, two searches scores highest (F1-only 0.378, F1+format 0.341, EM-only 0.386) and accuracy declines as the search count grows. The highest search-count bin sits at the evaluation turn budget (about five searches) and is almost entirely aborted (close to 100% of those rollouts reach the budget without closing an answer and score zero by construction), so the steep drop at the top of the curve is the truncation tail, not over-searching degrading a finished answer. The milder decline from two to four searches survives among completed rollouts but is confounded by question difficulty (a genuine four-hop question both needs more searches and is harder), so we read the curve as descriptive, not causal.

Second, and more informative, the curve is nearly identical across the three reward shapes: the reward shape does not change how well the policy answers at a given completed search count. What it changes is the termination rate. The fraction of held-out rollouts that abort (reach the turn budget or emit no parseable answer) is 22.8% for EM-only, 12.7% for F1-only, and 5.0% for F1+format, and EM-only places 22.0% of its rollouts in the fully-aborted top bin against 4.6% for F1+format (Figure D.11, right). EM-only's lower held-out exact-match is then better read as a termination gap rather than an answer-quality gap: when it finishes, it answers about as well as the F1 shapes, but it finishes cleanly far less often. This is the evaluation-time, per-question counterpart of the training dynamics in Section 4.3.4 (EM-only keeps 10 to 36% of training rollouts at the generation cap while the F1 shapes fall to about 1%) and of the valid-answer-rate channel in Section 4.1 (the F1+format advantage decomposing as format-adherence plus retrieval-grounding rather than chain-coherence). The analysis is correlational: the high search-count bins mix harder questions with the abort tail, the read is per-run at the best-EM checkpoint, exact-match is the benchmark's own scorer, and the search count is the number of search tool calls in the rollout transcript.

Table D.8: Completed-rollout exact-match by search count (the inverted-U of Figure D.11, left); the abort rate is over all bins. Pooled over three seeds and the four multi-hop benchmarks at each run's best-EM checkpoint.

# search calls	F1-only	F1+format	EM-only
1	0.248	0.254	0.272
2 (peak)	**0.378**	**0.341**	**0.386**
3	0.280	0.243	0.292
4	0.267	0.282	0.263
abort rate (all bins)	12.7%	5.0%	22.8%
mean search calls	2.69	2.64	2.87

E Cross-generation and paper-reproduction detail (backs Results 4.2)

The provenance behind Results 4.2. Two parts: (E.1) the paper-reproduction that validates the eval network the 0.8B numbers ride on; (E.2) the cross-generation floor-vs-trained table with its sources.

E.1 Paper targets vs our reproduction (Qwen2.5-3B Search-R1 GRPO)

Paper = Search-R1 v5 Table 3 (arxiv.org/html/2503.09516v5#A6); ours = our reproduction. Per-benchmark EM, paper vs our full-data reproduction (8x RTX 4090, 1 seed, IVF-SQ8):

Table E.1: Per-benchmark EM: Search-R1 paper targets versus our Qwen2.5-3B reproduction, for the base and instruct GRPO checkpoints.

Variant	NQ	Trivia-QA	PopQA	Hotpot-QA	2Wiki	MuSi-Que	Bam-boogle	avg EM
GRPO base, paper	0.421	0.583	0.413	0.297	0.274	0.066	0.128	0.312
GRPO base, ours	0.390	0.576	0.339	0.280	0.257	0.051	0.120	0.288
GRPO instruct, paper	0.397	0.565	0.391	0.331	0.310	0.124	0.232	0.336
GRPO instruct, ours	0.394	0.533	0.350	0.335	0.330	0.124	0.320	0.341

Both variants reproduce within ±2.5 pp of paper on avg EM (base 0.288 vs 0.312, −2.4 pp; instruct 0.341 vs 0.336, +0.5 pp), satisfying the "≤4 pp avg residual" criterion; every per-dataset residual is within 8 pp except instruct Bamboogle, which overshoots paper by +8.8 pp in our favour on n=125 (single-seed variance, not a shortfall). The eval network is therefore trustworthy for the 0.8B held-out numbers in 4.1.

E.2 Cross-generation: untrained floor + best trained, both generations

The Section 4.2 body table with its provenance. Untrained floor and best trained checkpoint, both measured on the same eval network:

Table E.2: Cross-generation untrained floor, best trained checkpoint, and RL lift on the same eval network, for Qwen2.5-3B and Qwen3.5-0.8B.

Model (gen)	Untrained floor	Best trained	RL lift	Floor source	Trained source
Qwen2.5-3B (older)	0.199	0.341	+0.142	raw Qwen2.5-3B-Instruct, full data	instruct GRPO
Qwen3.5-0.8B (current)	0.092	0.352	+0.260	terse-prompt untrained	F1-only, seed 44, step_310 (4.1)

Caveats (kept with the table): (i) the 3B floor 0.199 is the raw Qwen2.5-3B-**Instruct** (the comparator for the instruct GRPO endpoint); no separate base-untrained full-data floor was run. (ii) size is confounded with model generation (Qwen3.5 vs Qwen2.5). (iii) the two trained endpoints used different RL recipes (Search-R1 verl (Sheng et al., 2024) EM-only for 3B; our NeMo-RL (NVIDIA NeMo Team, 2025) reward-shape ablation for 0.8B), so this is "RL helps the small newer model most", not "our recipe beats the paper's"; anchor to our own 3B reproduction (0.341), not the paper rows in E.1. (iv) the released 3B checkpoint trained 1005 optimiser steps (per its released training script), not the 500 stated in the paper text; against our roughly one-epoch 0.8B run this widens, not narrows, the training-exposure gap, so the cross-generation reading is conservative.

F Training-health diagnostics (backs Results 4.3)

The six health + termination metrics behind the §4.3 body dynamics, per seed, three reward shapes overlaid (centred moving-average over the faint raw series); Figures F.1 to F.3 show the panels for seeds 42, 43, and 44. The collapse-signature deep read is folded in below.

What the six panels show (analysis):

- **Gradient norm:** bounded at every seed (~0.1 to ~1.3), no spikes, so optimisation never diverges. EM-only sits lowest, which is *under-training* from the sparse 0/1 reward (weak gradient), not extra stability. The only true divergence in the project was the 2x A100 **TP=2** run (the collapse note and Figure F.4 below).
- **Loss:** small and noisy around zero and decoupled from reward; under GRPO the policy-gradient loss is not a progress meter (read reward + entropy instead), so the flat-noisy loss is expected, not a stall.
- **KL penalty:** rises monotonically as the policy moves off the reference (beta=1e-3 keeps it gentle); the F1 shapes move further (more learning) than EM-only.
- **Approx. entropy:** declines for the F1 shapes (the policy sharpens) but stays pinned high for EM-only at seeds 42/43 (under-sharpening: sparse reward -> little gradient -> the policy never commits); only seed-44 EM-only sharpens. High entropy here is a symptom of weak signal, not healthy exploration.
- **Aborted ratio (no valid answer):** collapses from ~0.7 cold-start to ~0.01 within ~10 steps for the F1 shapes (the model learns to emit a parseable answer); EM-only stays elevated and re-rises late (seed-44 ~0.37 by the end; ~0.19 run-mean), the over-search-without-answering regime that explains its weaker held-out EM (Results 4.1).
- **Natural-termination rate:** the mirror image of aborted / length-clip (corr 0.93-0.99 in our runs), so it carries little independent signal here (aborts are ~99% truncations); ~1.0 for the F1 shapes, dips for EM-only. Kept logged because a different recipe could split malformed aborts from length-truncations.

Not charted (logged but uninformative): the **GRPO clamped probability-ratio (clip-ratio)** is inert at 1.0 every step. The run is fully on-policy (one optimiser step per generation batch), so at loss time the live and reference rollout policies are identical and the PPO clip never engages; the metric therefore carries no signal and is omitted from the panels. (Likewise turns/sample is just tool_calls + 1, corr 0.998, so it is dropped too.)

Cross-seed, the health story is stable in trend (same bounded-grad-norm, same F1-sharpens / EM-only-pinned entropy split at every seed); the divergent case is the TP=2 collapse only.

Read together: the collapse signature. The metrics are informative jointly, in the order they move: **entropy leads, grad-norm follows, reward dies last.** A reward shape that under-learns shows it first as pinned-high policy entropy (the policy never commits), then as a low gradient norm (sparse signal), and only last as a flat reward;

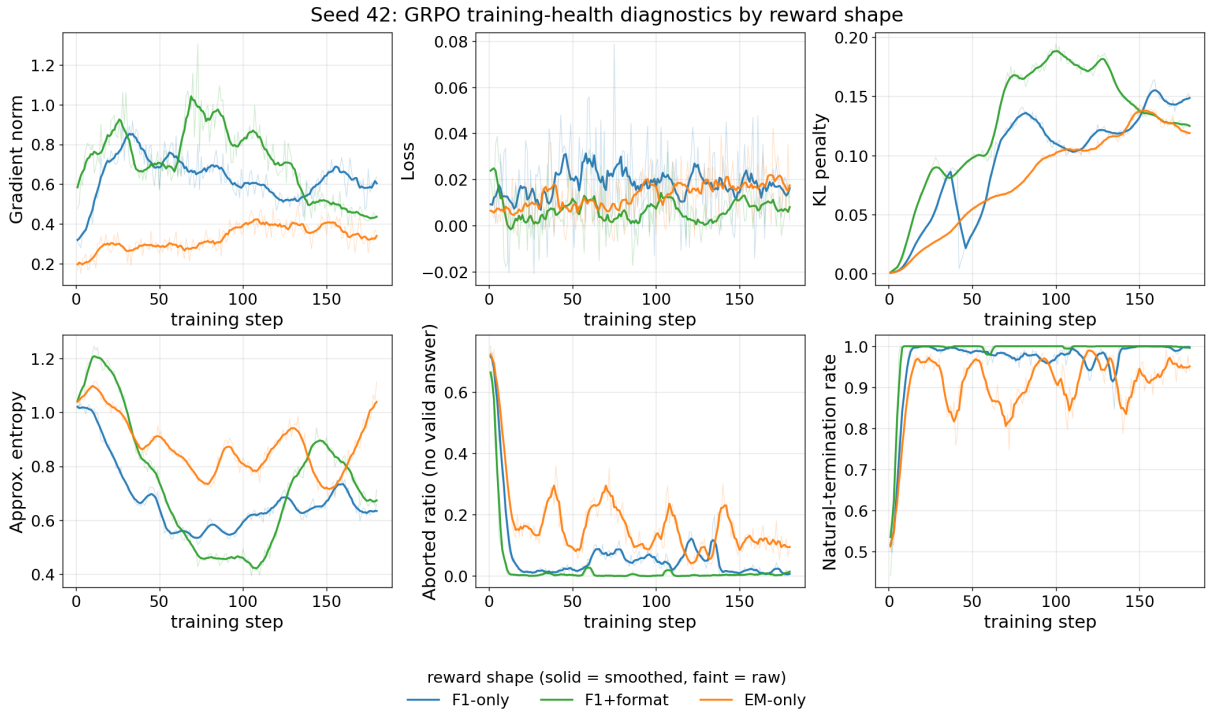


Figure F.1: Seed 42: gradient norm, loss, KL penalty, approx. entropy, aborted ratio, natural-termination rate.

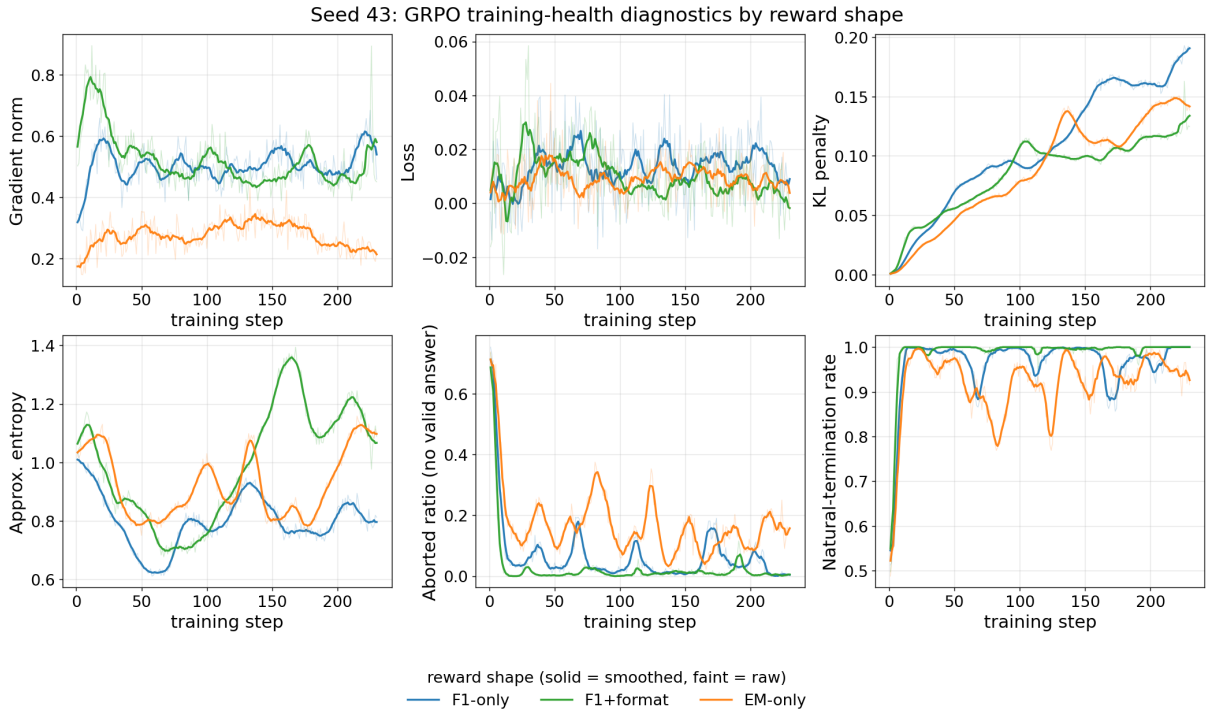


Figure F.2: Seed 43, same six panels.

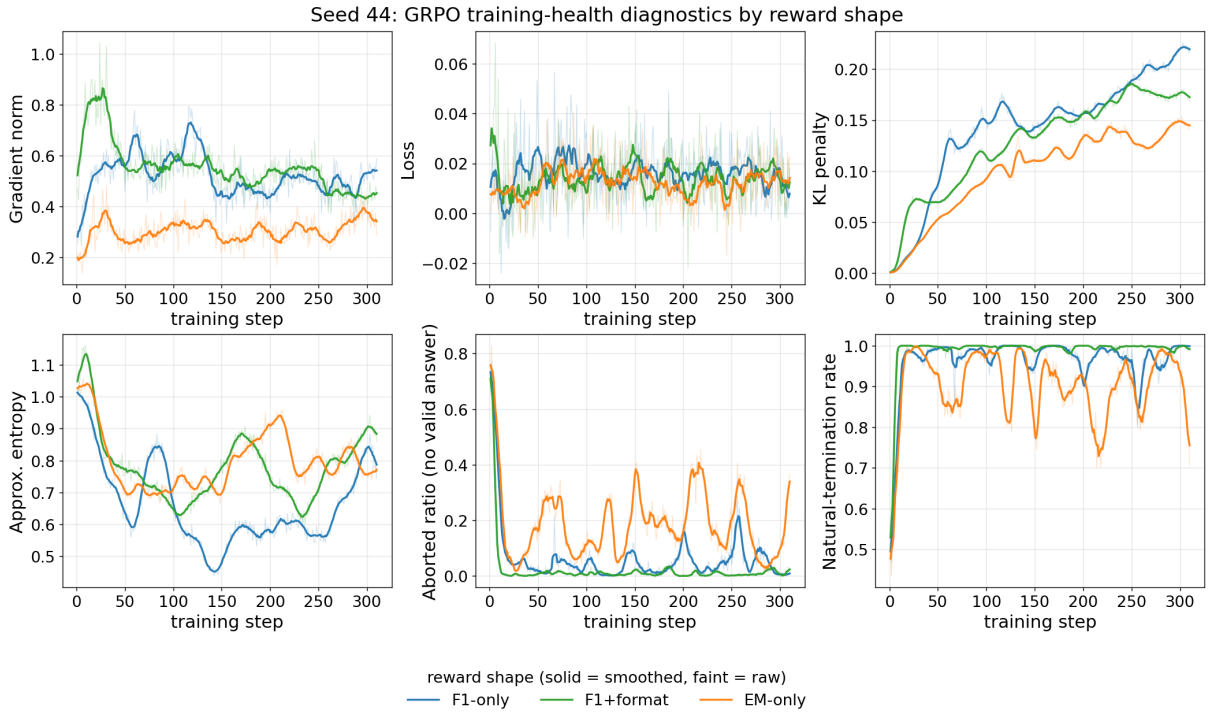


Figure F.3: Seed 44 (full epoch), same six panels.

this is exactly the EM-only signature at seeds 42/43. The healthy F1 shapes instead sharpen (entropy declines), drift further off the reference (KL rises), and keep a bounded grad-norm with no spikes. Watching reward alone would see trouble ~50 steps too late.

The one true divergence: a TP=2 mode collapse. Every data-parallel run stayed bounded; the only run that diverged was a 2x A100 **tensor-parallel (TP=2)** run (F1+format reward, seed 42). It trained healthily to ~step 200 (peak reward 0.391 @ step 170), then policy entropy collapsed and the gradient norm exploded (from ~0.7 to a peak of ~131 @ step 289); reward decayed and by step 313 the policy collapsed completely (reward 0, tool calls 0, 100% of rollouts truncated at the 1024-token cap), crashing at step 352. The same recipe and seed under DP=4 ran clean, so tensor-parallel sharding of the 248,320-token vocabulary is implicated, not the recipe (Figure F.4). This is why the whole study runs data-parallel (one full replica per GPU).

Still-climbing caveat. Several runs were length-bounded by budget and were still climbing at their final checkpoint (the full-epoch seed-44 legs in particular), so some headline numbers (including F1-only's seed-44 0.352) are conservative lower bounds, not converged ceilings.

F.1 Cadence-to-cadence stability (all nine runs)

Backs the trajectory-stability qualifier in Results 4.1.4. Each run's checkpoints were evaluated every 10 training steps (one cadence); the metric is the mean absolute step-to-step change in held-out avg EM between consecutive checkpoints (lower = calmer trajectory). Runs are sorted most-stable first.

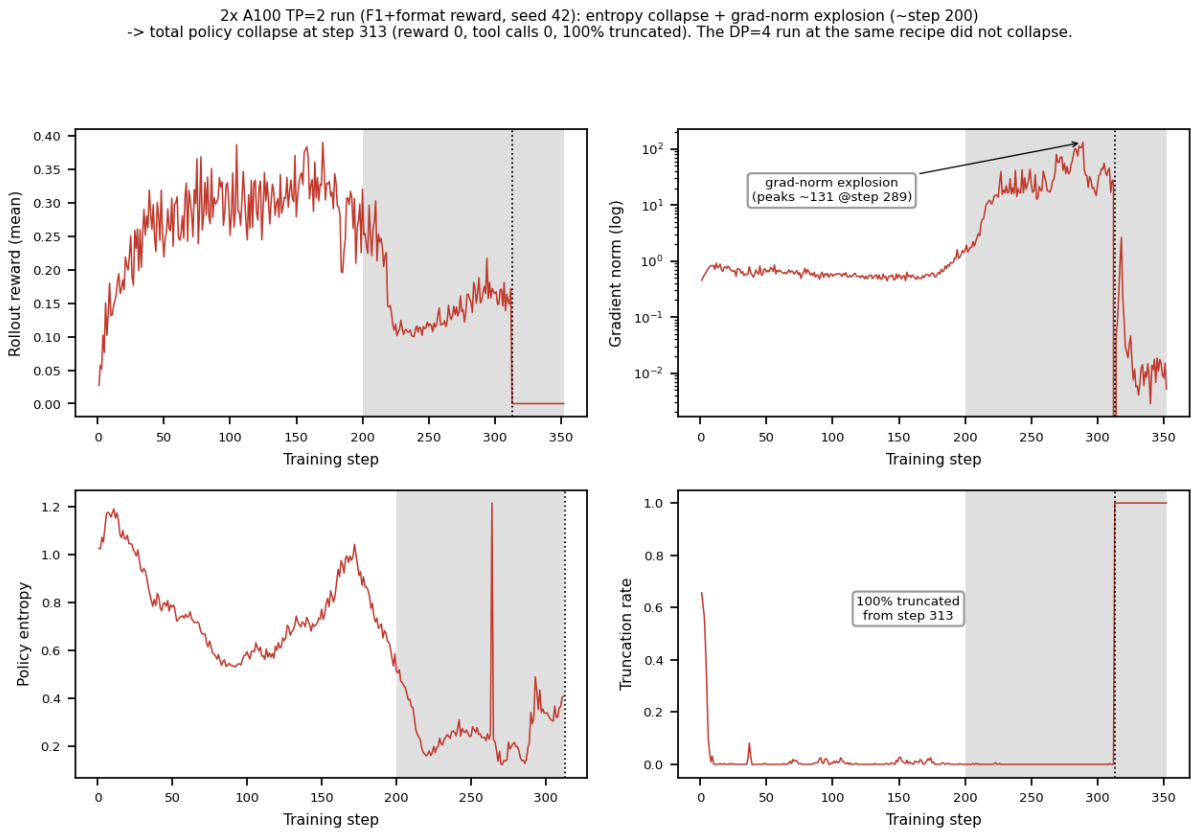


Figure F.4: The 2x A100 TP=2 run. Healthy to ~step 200, then entropy collapses and the gradient norm explodes to ~131 by step 289; reward dies to 0 by step 313. The DP=4 run of the same recipe and seed did not collapse.

Table F.1: Cadence-to-cadence stability of held-out average EM across the nine runs (mean and max absolute step-to-step change, std, and worst single drop), sorted most-stable first.

reward shape	seed	cadences	mean $\|\Delta\|$	max $\|\Delta\|$	std	worst single drop
F1+format	44	31	**0.015**	0.135	0.029	step 230->240 (-0.038)
F1-only	44	31	**0.016**	0.145	0.029	step 20->30 (-0.021)
F1+format	43	25	0.017	0.145	0.033	step 40->50 (-0.033)
EM-only	42	21	0.020	0.074	0.026	step 120->130 (-0.025)
F1-only	42	18	0.022	0.046	0.027	step 110->120 (-0.040)
EM-only	43	24	0.026	0.096	0.037	step 40->50 (-0.054)
EM-only	44	31	0.040	0.126	0.053	step 150->160 (-0.126)
F1+format	42	18	0.051	0.209	0.081	step 50->60 (-0.188)
F1-only	43	23	0.051	0.183	0.076	step 210->220 (-0.183)

Cadence-to-cadence stability of held-out avg EM, nine runs, most-stable first. The dominant lever is the seed, not the reward shape: the seed-42 F1+format run's 0.051 is 3.4x the seed-44 F1+format run's 0.015 under identical reward, driven by one outlier step (50 to 60). On top of that EM-only carries a secondary saw-tooth signature (its three runs rank 4th / 6th / 7th, and the seed-44 EM-only run is the noisiest despite the

calmest seed, from its -0.126 drop at 150 to 160). So "F1+format is the most seed-robust" (Results 4.1.2) holds for the best-checkpoint cross-seed spread, not necessarily the calmest trajectory.

F.2 Per-cadence reward-shape triad at seed 44 (C1-C31)

The cleanest reward-axis isolation in the corpus: same Qwen3.5-0.8B, same MuSiQue, same DP=4 Blackwell topology, same seed 44; only the reward shape differs. Each cell is the cadence-mean training reward / exact-match rate (%R=1; %R=1 = %EM for the EM-only run) over n=3200 rollouts per 10-step cadence. Bold marks each axis's run-high (reward and %R=1 are shown separately where they peak on different cadences). Backs the seed-44 same-seed reward-shape comparison and the ~20-22 % shared exact-match ceiling in Results 4.3.6.

Table F.2: Per-cadence training reward and exact-match rate (%R=1) for the three reward shapes at seed 44, isolating the reward axis.

Cadence	Steps	EM-only rew / %EM	F1-only rew / %R=1	F1+format rew / %R=1
C1	1-10	0.045 / 4.5 %	0.072 / 4.8 %	0.113 / 4.6 %
C2	11-20	0.078 / 7.8 %	0.121 / 7.4 %	0.188 / 5.9 %
C3	21-30	0.080 / 8.0 %	0.151 / 8.5 %	0.217 / 7.5 %
C4	31-40	0.115 / 11.5 %	0.164 / 9.8 %	0.240 / 10.2 %
C5	41-50	0.118 / 11.8 %	0.205 / 11.2 %	0.281 / 12.7 %
C6	51-60	0.131 / 13.1 %	0.207 / 12.4 %	0.264 / 11.9 %
C7	61-70	0.138 / 13.8 %	0.204 / 12.7 %	0.287 / 14.4 %
C8	71-80	0.150 / 15.0 %	0.242 / 15.6 %	0.302 / 15.2 %
C9	81-90	0.142 / 14.2 %	0.236 / 15.9 %	0.289 / 15.0 %
C10	91-100	0.155 / 15.5 %	0.256 / 17.5 %	0.303 / 15.9 %
C11	101-110	0.156 / 15.6 %	0.282 / 19.0 %	0.325 / 17.3 %
C12	111-120	0.146 / 14.6 %	0.245 / 15.4 %	0.315 / 16.6 %
C13	121-130	0.151 / 15.1 %	0.256 / 17.0 %	0.316 / 16.4 %
C14	131-140	0.147 / 14.7 %	0.279 / 19.4 %	0.325 / 17.2 %
C15	141-150	0.140 / 14.0 %	0.260 / 17.7 %	0.300 / 15.0 %
C16	151-160	0.155 / 15.5 %	0.299 / 20.4 %	0.335 / 18.9 %
C17	161-170	0.160 / 16.0 %	0.312 / 20.7 %	0.342 / 18.0 %
C18	171-180	0.153 / 15.3 %	0.292 / 19.7 %	0.329 / 16.9 %
C19	181-190	0.159 / 15.9 %	0.285 / 20.1 %	0.315 / 16.4 %
C20	191-200	0.165 / 16.5 %	0.287 / 19.7 %	0.331 / 17.5 %
C21	201-210	0.177 / 17.7 %	0.291 / 20.6 %	0.331 / 17.7 %
C22	211-220	0.160 / 16.0 %	0.301 / 21.1 %	0.342 / 19.0 %
C23	221-230	0.200 / 20.0 %	0.303 / 21.2 %	0.348 / 18.4 %
C24	231-240	0.196 / 19.6 %	0.281 / 19.2 %	0.328 / 17.3 %
C25	241-250	0.201 / 20.1 %	0.307 / **21.9 %**	0.350 / 19.9 %
C26	251-260	0.200 / 20.0 %	0.265 / 19.6 %	0.331 / 18.3 %

Cadence	Steps	EM-only rew / %EM	F1-only rew / %R=1	F1+format rew / %R=1
C27	261-270	0.171 / 17.1 %	0.291 / 19.7 %	0.332 / 16.8 %
C28	271-280	**0.216 / 21.6 %**	0.303 / 21.6 %	0.355 / **20.5 %**
C29	281-290	0.200 / 20.0 %	0.290 / 20.5 %	0.344 / 20.2 %
C30	291-300	0.209 / 20.9 %	**0.314** / 21.1 %	**0.358** / 19.9 %
C31	301-310	0.183 / 18.3 %	0.293 / 20.3 %	0.339 / 19.0 %

Per-cadence seed-44 reward-shape triad, single-axis isolation. All three shapes converge to a ~20-22 % %R=1 plateau (run-high spread only +1.4 pp: EM-only 21.6 % @C28, F1-only 21.9 % @C25, F1+format 20.5 % @C28). F1-only leads F1+format on %R=1 throughout (~1-3 pp) while F1+format leads on reward via the 0.1 floor; F1 leads EM-only through the early-to-mid run, and the two converge exactly at C28 (both 21.6 %), the cleanest signal that the exact-match ceiling at this scale is reward-shape-agnostic. C32 (partial, n=640, steps 311-312) omitted.

G Qualitative rollout traces

Three real rollouts (all from the F1-only run, seed 42), verbatim from the published training rollouts. Long retrieved chunks are elided with `[...]` and the fixed system/tool-schema preamble (Appendix B) is omitted; the questions, `<think>` blocks, `<tool_call>` queries, key returned snippets, and `<answer>` are quoted exactly. These are the full companions to the shortened examples in Results 4.3.

G.1 One broad search, right answer (reward 1.0) vs its failing sibling (reward 0.0)

Both rollouts are the same question at the same step (two members of the GRPO group), scoring 1.0 and 0.0; this best/worst pair is exactly the within-group contrast the advantage estimator pushes on. The instructive point: the winner does **not** cleanly resolve the language bridge. It fires a single broad search whose returned chunks happen to include the Kannada edition, and lands on the right winner despite confused reasoning. The loser mis-guesses the language (Bangla), retrieves only the Hindi edition, and returns the hosts instead of the winner. The shared failure mode is **bridge-entity brittleness** (resolving "language of *Aavarana*" -> Kannada), which persists across all three rewards.

```
# seed-42 F1-only, step 21, sample 0: reward 1.0  (3 of 5 returned chunks shown)
User: Who won season 5 of Bigg Boss presented in the language of the novel Aavarana?
<think> ... I need to ... search for information about Bigg Boss season 5 ...
particularly
since I should mention especially that the question asks about the language of the
novel
Aavarana. </think>
<tool_call><function=search><parameter=query> Bigg Boss season 5 winner
</parameter></function></tool_call>
<tool_response> "Bigg Boss (Hindi season 5)" ... won by ... Juhi Parmar ...
presented by Sanjay
   Dutt and Salman Khan ... </tool_response>
<tool_response> "Bigg Boss Kannada (season 5)" ... premiered on 15 October 2017 ...
rapper
   Chandan Shetty was declared the winner ... </tool_response>
<tool_response> [... 3 more chunks: "Bigg Boss" (general), "Bigg Boss Kannada",
"Bigg Boss
   (Hindi season 6)" ...] </tool_response>
<think> ... Bigg Boss Kannada (which the user asked about since they mentioned the
language)
   ... Chandan Shetty was declared the winner ... However, the user specifically
asked about
   the language of the novel Aavarana, which means they're asking about Bigg Boss in
 Hindi in
```

```
   that language ... season 5 in the Kannada language would be the one to look for.
</think>
<answer> Chandan Shetty </answer>      # correct (reward 1.0), via one broad search
+ confused bridge reasoning
```

```
# seed-42 F1-only, step 21, sample 1: reward 0.0  (the failing sibling)
User: Who won season 5 of Bigg Boss presented in the language of the novel Aavarana?
<think> ... Let me start by searching for Bigg Boss season 5 in Bangla (Aavarana
language). </think>   # wrong: Aavarana is Kannada, not Bangla
<tool_call><function=search><parameter=query> Bigg Boss season 5 bangla
</parameter></function></tool_call>
<tool_response> [... 5 chunks, all "Bigg Boss (Hindi season 5)" / "Bigg Boss"
(general);
   hosts Sanjay Dutt and Salman Khan ...] </tool_response>
<think> ... for Bigg Boss season 5 in Hindi, the hosts were Sanjay Dutt and Salman
Khan.
   There's no evidence in the search results that the show existed in Bangla/Avarana
 language ... </think>
<answer> Sanjay Dutt and Salman Khan </answer>    # wrong (hosts, not winner; Hindi
default), reward 0.0
```

G.2 Explicit numbered planning, clean 3-hop chain (reward 1.0)

The second learned pattern: the policy writes an explicit numbered plan, issues one targeted query per hop, and substitutes the resolved entity between hops. This is the chain-of-search behaviour the recipe is meant to teach.

```
# seed-42 F1-only, step 39, sample 158: reward 1.0
User: How many floors are in the Trump Tower in the city where the performer of Come
 2 My House was born?
<think> ... Let me break this down: 1. First search to find information about "Come
2 My House"
   performer and where they were born; 2. Then find the location of Trump Tower in
that city;
   3. Finally, determine the number of floors in Trump Tower. </think>
<tool_call><function=search><parameter=query> Come 2 My House performer birthplace
information </parameter></function></tool_call>
<tool_response> "Come 2 My House" ... ninth studio album by American R&B/funk singer
 Chaka Khan ... </tool_response>   # performer found, no birthplace
<think> ... found that "Come 2 My House" performer is Chaka Khan ... I need to find
out where
   Chaka Khan was born. </think>
<tool_call><function=search><parameter=query> Chaka Khan city of origin or birth
place </parameter></function></tool_call>
<tool_response> "Chaka Khan" ... was born ... on March 23, 1953 ... in Chicago,
Illinois ... </tool_response>
<think> ... Chaka Khan ... was born in Chicago, Illinois. Now I need ... Trump Tower
 in Chicago ... </think>
```

```
<tool_call><function=search><parameter=query> Trump Tower location Chicago Illinois
</parameter></function></tool_call>
<tool_response> "Trump International Hotel and Tower (Chicago)" ... built the
98-story structure ... </tool_response>
<tool_response> "Trump Tower" ... a 58-story ... skyscraper ... in Midtown
Manhattan, New York City ... </tool_response>
<think> ... conflicting: 98-story (Chicago) vs 58-story (Manhattan) ... Since she
was born in
   Chicago, I'll use the 98-story figure from the Chicago location. </think>
<answer> 98 </answer>      # correct, reward 1.0
```

The tell is the last hop: the model resolves the Chicago-vs-New-York ambiguity from the chain context (the birthplace found in hop 2), not by guessing the more famous tower.

G.3 Cold-start failure (reward 0.0)

At cold start the format is already perfect but entity resolution is not. The policy never resolves Coloane -> Macau (it confabulates a UK location), and answers a non-existent phrase, even though the actually-relevant civil-law chunk was retrieved. (Most cadence-1 rollouts also run long: ~93 % of step-1 rollouts approach the 8192-token cap; this one instead commits a wrong answer after one search.)

```
# seed-42 F1-only, step 1, sample 0: reward 0.0
User: What tradition is the basis for the legal system in the place where Coloane
Park is located?
<think> ... Let me search for Coloane Park and the American legal system/traditions.
 </think>   # never resolves Coloane -> Macau
<tool_call><function=search><parameter=query> Coloane Park legal system tradition
</parameter></function></tool_call>
<tool_response> "Beating the bounds" [...] </tool_response>
<tool_response> "Ceffyl Pren" ... a traditional Welsh form of mob justice ...
</tool_response>
<tool_response> "Civil law (legal system)" ... originating in Europe ... codified
... </tool_response>   # the actually-relevant chunk
<tool_response> "Kissing traditions" ... custom in the Castle of Blarney in Ireland
... </tool_response>
<tool_response> "Colloquy (law)" [...] </tool_response>
<think> ... Coloane Park is in the United Kingdom ... a public park in Welham, near
Hounslow ...
   the UK has a tradition called "Kissing traditions" ... Wait, that doesn't match
what I got.
   Let me re-examine. ... the Cefyl Pren ... </think>
<answer> Kissing Traditions </answer>     # confabulated; ground truth =
Portuguese-influenced civil law (Macau). reward 0.0
```

The model even retrieved the correct civil-law chunk but never connected it, because it placed Coloane Park in the UK rather than Macau: the failure is entity resolution, not retrieval coverage.

H Pilot study (Qwen3-0.6B) full results

The pilot that fixed the main-study design decisions (summarised in Method 3.8). 29 runs across two tag-format blocks (14 + 15). Three findings carry into the main study, each shown below: the 0.1-floor / prompt-sensitivity result (H.1), the base-model bootstrap failure (H.3), and the in-distribution-vs-out-of-distribution tag cost (H.4). These three are the decisive pilot graphs; the trained-vs-untrained eval (H.2) is the fourth.

H.1 Prompt sensitivity and the 0.1-floor finding

Three Qwen3-0.6B (Qwen Team, 2025) runs identical in config and reward, differing only in the user prompt: Run X (a baseline instruction with one worked example) settled at 2.0 tool calls / reward 0.190; Run Y (a rephrased instruction, one example) at 1.0 call / 0.189; Run Z (the rephrased instruction, no example) at 0 calls / 0.159. Two findings, both decisive for the main study: (a) tool-using and no-tool runs are separated by only 3 to 6 pp of reward, because the ReSearch **0.1 floor already pays a parseable-but-wrong answer** (this is what motivated trying F1-only); (b) prompt phrasing alone moved the converged tool-call rate from 0 to 2 at constant config, so at 0.6B the **prompt is the dominant behavioural lever** (this is why the Qwen3.5 prompt was fixed by its own ablation, I, before the reward ablation).

H.2 Trained vs untrained eval (two 0.6B checkpoints)

Two trained 0.6B checkpoints were each evaluated against the untrained hybrid on the full held-out suite (51,713 rows per variant). We report them as two separate trained-versus-untrained results, not as a comparison between the two checkpoints: they differ



Figure H.1: Three runs identical except the user prompt (X/Y/Z). Tool use settles at 2 / 1 / 0 calls per episode (centre) while reward converges to ~0.16 to 0.19 (left), every run above the 0.1 format floor (dashed).

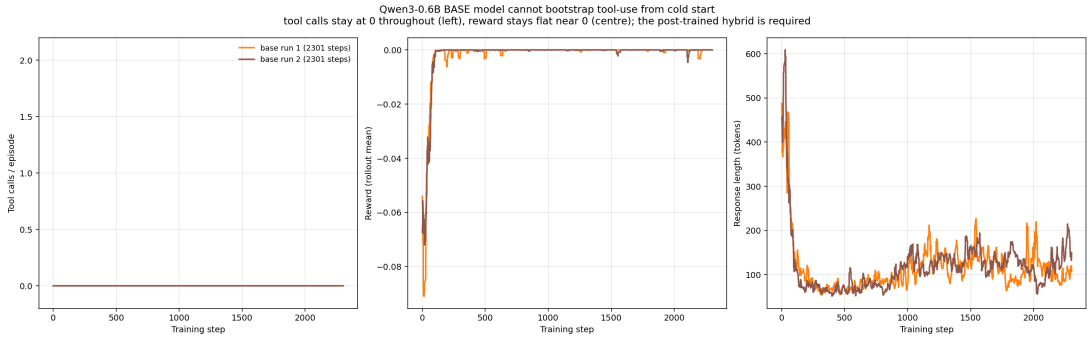


Figure H.2: The two base-model runs that trained the full 2301 steps: tool calls stay at 0 (left) and reward stays flat near 0 (centre). The two other base runs stopped early and are omitted for legibility.

in prompt, behavioural regime, and training length, so no cross-checkpoint attribution is possible.

- **The with-example, heavy-tool checkpoint:** avg EM 0.102 -> 0.155 (+0.053, +52%); 6/7 datasets up on EM, 2Wiki tied (-0.003, capacity-bound at 0.6B); all 7 up on ACC and F1. Held-out, so this rules out memorisation: GRPO over the small hybrid produces a real generalising lift even at 0.6B.
- **The no-example, decision-rules, leaner checkpoint:** avg EM 0.169 over the same untrained baseline, reached with ~1100 response tokens per question, a low tool-call rate, and a training reward of 0.215.

What the two evals taught us, carried into the main study: (a) the untrained-to-trained lift is real and held-out, so the within-model-lift logic (3.1) was first validated here; (b) a terse, no-example prompt regime is compatible with a strong held-out score at a low token and tool budget, which pointed the main-study prompt design toward the terse no-example form (Appendix I).

H.3 Base model cannot bootstrap

All four base-model attempts stayed at 0 tool calls throughout, with flat near-zero reward and short (~110 to 125-token) responses: the base (non-post-trained) model never emits the tool-call structure at all. This is why the main study trains the **hybrid** (the analogue of the papers' Instruct models), and why the post-trained prior is called decisive in §4.2.

H.4 In-distribution tags are free at matched step count

The pilot v0 block used the papers' `<search>`/`<result>` tags; v1 switched to in-distribution `<tool_call>`/`<tool_response>` JSON. The often-quoted ~4 pp gap (best v0 0.215 vs best v1 0.179) is a **training-length artifact, not a tag-format cost**: v0's 0.215 is at 2279 steps and v1's 0.179 at 884. Cut v0 to the same 884-step horizon and the two converge to within noise (SMA-20 reward 0.169 out-of-dist vs 0.191 in-dist). So the stricter native-JSON validation surface costs nothing once the step budget is matched,

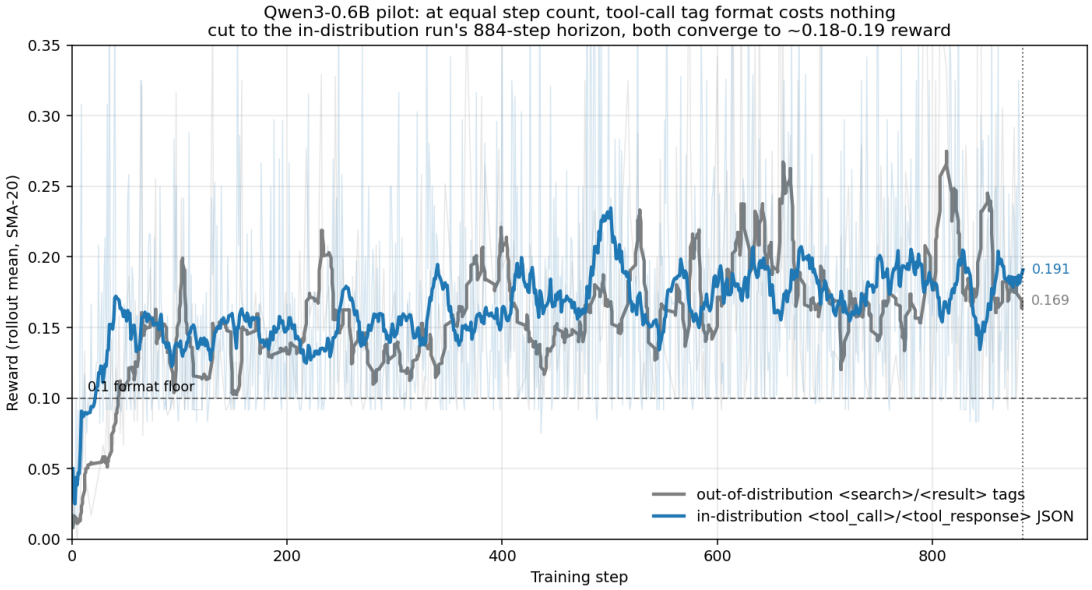


Figure H.3: The two best-of-block pilot runs. Out-of-distribution `<search>`/`<result>` (grey) and in-distribution `<tool_call>`/`<tool_response>` JSON (blue) converge to nearly the same reward (~0.17 to 0.19), both above the 0.1 floor (dashed); the out-of-distribution run only reaches its quoted 0.215 ceiling by training far longer.

which is what lets the main study use the Qwen3.5-native tags for train/eval alignment at no reward penalty.

I Prompt-mode ablation (selecting the terse prompt)

The terse 170-char user prompt (B) was fixed in **two ablation steps** on subsamples of the 7 benchmarks, before the reward ablation, so that all three reward runs share one prompt and the only moving variable is the reward.

- **Step 1 (minimal vs verbose; smoke, n=100/dataset):** dropping a verbose system-protocol default (~330 words before the question, mean EM 0.0086) for a minimal user-message form lifted untrained-hybrid EM to 0.057, a **6.6x** lift. This is where the 6.6x belongs.
- **Step 2 (12-variant selection; n=300/dataset across all 7 benchmarks):** a single-knob ablation over 12 prompt variants then selected the terse variant at mean EM 0.103, **+0.0436 over the n=1000 minimal baseline 0.0594 (~1.7x)**, all 7 datasets positive (sign test p=0.0078); the next-best variant (a research-role phrasing) reached 0.085, and the other ten failed the +0.025 bar.

The terse lock was then validated at full data (51,713 rows) at mean EM **0.092**, the untrained floor the trained model is measured against (§3.1). Attribution stays clean: the 6.6x is the step-1 minimal-vs-verbose lift, not terse's own; terse's contribution on top of minimal is the ~1.7x. The finding is asymmetric: the hybrid **needs** the auto-injected `# Tools` + `<IMPORTANT>` block (in-distribution), while the base model needs it stripped.

J Chain-flip audit (LLM judge)

A targeted audit of whether perfect-reward (reward=1.0) rollouts are actually coherent, or whether the policy reaches the right answer through a **silent entity flip** in its `<think>` chain (a reward-hack the EM/F1 reward cannot see). This separates an "the answer is right" claim from an "the reasoning is right" claim, on the seed-42 F1-only and F1+format runs.

Why an LLM judge, not regex. Two code-based detectors were built and both failed validation, which is the entire reason an LLM judge was needed: the regex detectors had near-zero precision (0 of 121 hand-labelled flags were real silent flips; the +9.1 pp "signal" was a junk-bridge artifact) and, once sharpened for precision, low recall (the sharpened form misses phrasings like "the country containing Fox Island is ..."). So the regex route is unusable; only the LLM-judge result is trustworthy.

Process. The judge is Qwen3.5-35B (the largest dense Qwen3.5 that fits the 2x4090) served via Ollama (`/api/chat`, `num_ctx=16384`, `think:false`), reading the full `<think>`/`<tool_response>` chain. It was calibrated on a 157-trace hand-labelled set (132 clean / 24 ambiguous / 1 flip): the production configuration (the 35B judge with the final judge prompt, reproduced below) reached 7.6% false-positive on the hand-clean set and 77.7% three-way agreement, after a 27B-Q4 judge plateaued at 39% FP (capacity gap dominated the prompt gap). The audit set was 2,000 perfect-reward rollouts (1,000 F1-only + 1,000 F1+format), cadence-stratified, Wilson 95% CI ~+-1.5 pp.

Results. At n=1091/2000: F1-only silent-flip rate **2.38%** (13/546) vs F1+format **4.04%** (22/545), a -1.66 pp gap, direction stable across four sample points (n=749 to 1091). F1+format has **more** silent flips than F1-only, not fewer, which is the **opposite** of the earlier regex-era prior. The reading: F1+format's held-out EM lead over F1-only at seeds 42/43 comes **despite** slightly worse chain coherence; the format gate + 0.1 floor rescue incoherent chains by enforcing answer format, not by improving the reasoning. This strengthens the §4 attribution decomposition (format-adherence and retrieval-grounding are the decisive channels; chain coherence is a small negative contributor for F1+format). A spot-check of the judge's flagged traces found 2 of 10 apparent false positives were actually real flips the regex-scoped hand audit had missed, so the true FP rate is likely $\leq 3\%$. **Lesson on small-n audits:** an n=171 prefix gave the prior-confirming direction; ~4x more data flipped it (the direction reversed at n=749), so interim conclusions on <500 traces are unsafe when the true rate is low single-digit.

The judge prompt (instruction block). The production judge prompt is reproduced below, ASCII-normalised (typographic dashes and arrows shown as `--` and `->`). Five worked examples (one real flip, one ambiguous, three clean) follow this block in the deployed prompt as few-shot demonstrations.

```
You audit multi-hop question-answering reasoning chains for ONE specific
  failure mode: the SILENT ENTITY FLIP. Be extremely conservative --
  most chains are NOT flips. The base rate is ~1-5% flip, ~95-99% clean.

A multi-hop question is answered by resolving intermediate "bridge
```

```
  entities" (e.g. "the country containing Fox Island", "the director of
  film X"). Each bridge entity has a ROLE (what it is) and a VALUE (the
  model's answer for it).

A SILENT ENTITY FLIP requires ALL of the following:
1. The SAME role gets two different values across the chain.
2. The chain COMMITS to value A (states it as the answer to the
  sub-question, not just considers it).
3. The chain LATER COMMITS to value B for the SAME role.
4. NO search result between the two commitments establishes value B.

If ANY of the four conditions fails, the chain is NOT a flip.

Classify each chain as exactly one label:

- "clean": no silent flip. This includes ALL of the following common
  cases that LOOK like flips but are NOT:
  (a) Different roles in different hops (e.g. "country containing X" in
    hop 1, "city where Y happened" in hop 2 -- these are different
    roles, not the same).
  (b) Exploration of multiple candidates from search results (e.g.
    search returns "Mountain Meadow Ranch" and "Mountain Ranch,
    California" -- the chain considering both before committing is NOT a
    flip).
  (c) Variant names / abbreviations / partial captures for the same
    entity ("Philly" vs "Philadelphia", "St." vs "Saint", "DC" vs
    "Washington DC", "U.S." vs "United States" -- these are the SAME
    value, not a flip).
  (d) Two different proper-named entities that share part of a name
    treated as separate (e.g. "Petronius Arbiter" vs "Sextus Claudius
    Petronius Probus" are DIFFERENT people; the chain mentioning both is
    exploration, not a flip).
  (e) Any value change AFTER a relevant search result -- the search
    result JUSTIFIES the change even if the search result is wrong or
    only partially on-point.
  (f) Wrong-but-consistent reasoning (model picks a wrong value and
    sticks with it the whole chain).
  (g) The chain notices an error and corrects itself, with the
    correction visible in the same or next THINK block.

- "flip": ALL four conditions above hold. The same role goes from a
  committed value A to a committed value B with no intervening search
  result that could plausibly justify the change.

- "ambiguous": the chain is too muddled to classify, OR there is
  borderline evidence of a possible flip but the chain is also doing
  exploration that could explain it.

CRITICAL RULES (in priority order):
1. Default to "clean" when in doubt. Most chains ARE clean (base rate
  ~95%).
2. The role must be IDENTICAL across both values for it to be a flip.
  "country containing Fox Island" and "city of Pan-African Conference"
  are DIFFERENT roles, not a same-role flip -- even if both end up in
  the FINAL ANSWER sentence.
3. ANY plausibly-relevant search result between value A and value B
  makes it clean. Do not second-guess whether the search result is "good
```

```
  enough" -- its presence is enough. If you find yourself writing
  "search results confirming X appear between these thoughts BUT..." --
  stop. The "but" is wrong. Label it clean.
4. Variant spellings, abbreviations, and synonyms (St / Saint, Philly /
  Philadelphia, US / USA / United States) are the SAME value.
5. Two different real-world entities that share part of a name are
  DIFFERENT values -- the chain naming both is exploration, not
  flipping.
6. Self-correction within the next THINK block (model says wrong thing,
  then immediately retracts within 1-2 sentences) is clean, NOT a flip.

ANTI-PATTERNS -- these LOOK like flips but are NOT. Do not label these
  "flip":

  - "THINK 2 commits to A; THINK 5 commits to B for the same role.
    Although a search result mentioning B appears between THINK 2 and
    THINK 5..." -> CLEAN. The search result justifies the change. Stop
    here.

  - "THINK 3 identifies State_X as the answer to sub-question 1; THINK 5
    uses City_Y from State_Z as input to sub-question 2." -> CLEAN.
    Sub-question 1 and sub-question 2 have DIFFERENT roles.

  - "THINK 4 says 'California is in the United Kingdom' then in the next
    sentence corrects to 'California is in the United States'." -> CLEAN
    (self-correction).

  - "Chain considers Candidate_A from search result 1 and Candidate_B
    from search result 2, then commits to Candidate_A." -> CLEAN.
    Exploration of search candidates is not a flip.

  - "Chain mentions Person_X (one bridge entity) and Person_Y (different
    bridge entity) where their names overlap partially (e.g. both are
    'Petronius')." -> CLEAN. They are different people, not the same
    role with two values.

If you cannot point to a SPECIFIC sentence where the chain COMMITS to
  value A as the answer to sub-question R, AND another SPECIFIC sentence
  where the chain COMMITS to value B as the answer to the SAME
  sub-question R, AND NO search result mentions B between them, then it
  is NOT a flip.

Respond with ONLY a JSON object, no other text:
{"label": "clean|flip|ambiguous", "bridge_roles": ["role = value, role =
  value, ..."], "flipped_role": "the role that flipped, or null",
  "explanation": "one or two sentences"}
```

K Public artifacts

The held-out numbers in Results 4.1 ride on an evaluation pipeline whose fidelity rests on the paper reproduction in Appendix E.1 (within ±2.5 pp) and train/eval byte-alignment (A.2). The public artifacts are listed below.

Public artifacts. All artifacts are public: the 9 training-checkpoint repositories, the unified held-out-eval dataset (`pantomiman/reason-over-search-eval-m5`; all 1,554 checkpoint x benchmark metric cells), the 9 unified training curves on Weights & Biases, and the source code (`GaurisankarJ/r_o_s`).

L Compute and cost

This is the **only** place compute and cost are reported; the main body makes no compute-budget claim. Training ran on a **4x RTX PRO 6000 Blackwell** node (96 GB per GPU) provided through a **CloudRift compute grant**, in a data-parallel **DP=4** topology (one full Qwen3.5-0.8B replica per GPU) with a colocated vLLM rollout engine; all runs are reported uniformly on this setup (Appendix A).

Per-run budget. A full-epoch run (~312 steps on the MuSiQue training set) takes ~**29 h wall-clock** at the DP=4 pace (~5.6 min/step), roughly ~**115 GPU-hours**; the partial-epoch legs (~210 to 251 steps) take proportionally less.

Grant and total cost. Training compute was a **CloudRift grant of $2,000**, enough for about **14 days** on the 4x Blackwell node. This study used roughly half of it: approximately **$700 to $800 over ~7 days** for the reward-shape x seed training runs and the held-out evaluation performed on this node (the evaluation corpus totals 1,554 checkpoint x benchmark cells across all legs). A single full-epoch run (~29 h) is therefore on the order of ~**$120 to $140** of the grant; the partial-epoch legs less.