Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

# Post-Training on the Clock: Specializing a Small Language Model on a Consumer GPU

**Uygar Kurt**

uygarpub@gmail.com

Independent Researcher https://orcid.org/0009-0001-1861-2115

**Research Article**

**Keywords:**

**Posted Date:** November 2nd, 2026

**DOI:** https://doi.org/10.21203/rs.3.rs-11540864/v1

**License:** This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

**Additional Declarations:** The authors declare no competing interests.

# POST-TRAINING ON THE CLOCK: SPECIALIZING A SMALL LANGUAGE MODEL ON A CONSUMER GPU

**Uygar Kurt**

**ABSTRACT**

How should a language model be specialized for a target task when training is limited to one consumer GPU and a few hours? We study the allocation of a nominal three-hour training budget, with up to 0.3 additional hours, between supervised fine-tuning (SFT) and Group Relative Policy Optimization (GRPO). We use tool calling as a case study because of its practical relevance to language-model agents and its verifiable outputs. Using Qwen2.5-0.5B-Instruct with low-rank adaptation on an RTX 5060 Ti with 16 GB of memory, we compare SFT, GRPO, and six sequential SFT-to-GRPO allocations. Training uses a filtered mixture of xLAM function-calling examples and Hammer irrelevance examples, covering both call construction and abstention when no offered tool is suitable. Evaluation reports xLAM exact tool-call and function-name accuracies for local call construction, BFCL-v3 Single-turn FC (agg) for external function calling, BFCL-v3 Irrelevance (agg) for abstention, and MMLU, MATH-500, and IFEval for selected general capabilities. The suffix (agg) denotes study-specific averages of selected subset accuracies weighted by subset size. At the last evaluated checkpoints within 3.3 reported hours, SFT achieves the highest xLAM exact tool-call accuracy (82.82%), compared with 69.13% for GRPO alone. Selected hybrid allocations improve external function-calling accuracy over SFT, with different abstention trade-offs. Most sampled GRPO groups have constant rewards and provide no relative task-reward signal. We release the prepared mixed dataset, all allocation-study checkpoints, and Post-Training Lab, an editable training and evaluation codebase with CUDA and MLX implementations.

## 1 Introduction

Specializing a language model for a target task under a short training budget requires choosing an adaptation recipe. A practitioner must choose between supervised fine-tuning (SFT), which learns from reference responses; reinforcement learning (RL), which learns from rewards on generated responses; or a sequential combination of the two. We study this choice using Group Relative Policy Optimization (GRPO) [26] as the RL method.

We use tool calling as a case study because of its practical relevance to language-model agents and its verifiable outputs [16, 34]. Structured tool calling maps a user query and offered functions to correct calls or abstention when none is suitable. Small specialized models can make this capability accessible on consumer hardware.

We specialize Qwen2.5-0.5B-Instruct [22] on one NVIDIA RTX 5060 Ti with 16 GB of memory. All reported runs use low-rank adaptation (LoRA) [6] to reduce trainable-parameter and optimizer-memory costs under this resource constraint. The released training scripts also support full-parameter fine-tuning. Six intermediate checkpoints from a shared SFT trajectory initialize GRPO continuations, alongside SFT-only and GRPO-only baselines. We use a nominal three-hour training budget with up to 0.3 hours of headroom. The primary comparison selects each trajectory's last evaluated checkpoint at or below 3.3 reported hours. Hybrid totals include their SFT initialization time.

Our training data combine xLAM function-calling examples [16, 34] with the irrelevance dataset released by Lin et al. [13]. The original xLAM corpus is *positive-only*: every reference answer contains at least one function call. It therefore provides no examples of when to abstain. Hammer supplies complementary examples with empty reference call lists, constructed by removing the appropriate functions from offered tool lists. We mix the sources and remove rows

with ambiguous duplicate function names and negative examples with no offered tools. This preserves the distinction between choosing no call from an available tool list and ordinary interaction without tools. We release this dataset.

Our contributions are:

1. **A consumer-hardware recipe comparison.** We compare SFT, GRPO, and six sequential allocations under the stated training-time budget, reporting call construction, abstention, and selected general-capability metrics separately.
2. **An editable research implementation.** We release Post-Training Lab [1] with explicit training objectives, LoRA and full-parameter CUDA training, TensorBoard diagnostics, local dataset evaluators, and an additional native MLX backend. The experiment branch [2] contains the implementation used in this study as well as the evaluation scripts.
3. **Released dataset and checkpoints.** We release the prepared mixed xLAM [3] and all 40 allocation-study training checkpoints [4], with source attribution, filtering and split procedures, and reproducible training configurations.

## 2 Related work

Tool-calling research has expanded what models must learn: when to invoke an application programming interface (API), which function to select, and how to supply its arguments. Toolformer learns API insertions from self-supervised data [25], while ToolLLM builds instruction-tuning trajectories over large API collections [21]. APIGen emphasizes automated generation and verification [16]; xLAM develops action models from such data [34], and ToolACE broadens coverage through agent-based synthesis [15]. These efforts provide reusable data for specialization, while leaving the choice of adaptation recipe open.

Learning when to abstain is part of this problem. Although the public xLAM 60k corpus contains positive calls, APIGen augments its own model training with irrelevance and insufficient-argument examples [16]. Lin et al. explicitly study over-calling after positive-only training and add irrelevant-tool examples alongside function masking [13]. When2Call further distinguishes calling, requesting clarification, and explaining that the available tools cannot answer [24]. Our mixture adapts existing negative data, with reproducible filtering and separate call-construction and abstention evaluation.

Verifiable calls also enable direct RL feedback. ToolRL uses graded rewards for format, tool selection, and arguments [20]; Nemotron-Research-Tool-N1 instead combines format and call correctness in a binary reward [36]. Both report RL gains over SFT baselines under their respective recipes. SLCA-GRPO further examines how rewards are assigned to tool-call and summary segments [33]. Laskar et al. [10] directly compare SFT, GRPO, and SFT-to-GRPO across six Qwen3 scales from 0.6B to 32B: LoRA-SFT is strongest in most in-distribution settings, while GRPO has a modest cross-dataset transfer advantage. Laskar et al. do not match training compute across methods, leaving open how these methods compare under a shared, short time budget.

The supervised starting point also affects what subsequent RL can learn. In mathematical reasoning, Li et al. [11] find that response diversity can help identify better starting checkpoints for RL, while Kang et al. [8] find that generalization measures can predict subsequent RL performance more reliably than SFT accuracy alone. SRFT combines supervised and reinforcement objectives within one stage [1], extending the design space beyond sequential training. We compare several SFT prefixes along one shared trajectory; our allocation study covers those sequential recipes rather than all possible schedules.

Comparing recipes requires specifying the resource being allocated. Raghavendra et al. [23] divide annotation budgets between supervised and preference-based tuning. Wang et al. [31] characterize near-optimal allocation regions and their transfer across model scales, primarily under annotation budgets, with an additional GPU-time case study for SFT and direct preference optimization. Vattikonda et al. [30] already branch RL from shared SFT checkpoints and analyze frontiers measured in floating-point operations for multi-step web agents.

Under a short deadline, the information obtained per rollout matters as well as its computational cost. Equal rewards give zero group-relative task advantages by construction. DAPO and gradient-starvation work diagnose the resulting inefficiency and propose sampling or advantage-estimation remedies [18, 32]. RC-GRPO addresses low within-group reward variation in multi-turn tool calling through reward-conditioned sampling [38]. Liu et al. [14] analyze wasted tool-calling rollouts and update costs, comparing acceleration under equal wall-clock time.

[1] `https://github.com/uygarkurt/post-training-lab/tree/main`

[2] `https://github.com/uygarkurt/post-training-lab/tree/time-constraint-trianing-research`

[3] `https://huggingface.co/datasets/uygarkurt/xlam-tool-calling-mixed`

[4] `https://huggingface.co/collections/uygarkurt/post-training-on-the-clock`

Cramming [3] studies language-model pretraining from scratch on a single GPU within one day. We similarly evaluate performance under fixed hardware and training-time constraints, focusing on post-training.

Existing systems support model-parallel training (Megatron-LM [28]), RL post-training (verl and OpenRLHF [7, 27]), and fine-tuning/data workflows (LlamaFactory and ActionStudio [35, 37]). Post-Training Lab complements these systems with explicit, editable objectives, CUDA and native MLX implementations, and the prepared data, evaluators, and checkpoints accompanying this measured comparison.

## 3 Study design

### 3.1 Data, filtering, and abstention

We prepare the dataset using APIGen/xLAM and MadeAgents/xLAM-irrelevance, used to train the Hammer series of models. MadeAgents/xLAM-irrelevance constructs negatives by removing the reference function from sampled xLAM tool lists and setting the target to `[]`. We reuse these examples without generating new data or applying Hammer's function masking.

The positive-only control motivates including negatives: after 2.7 reported hours of SFT, BFCL-v3 Irrelevance (agg) falls from 66.93% to 7.22%, indicating failure to abstain when no offered tool is suitable. Mixed SFT at the same time scores 67.47%, with similar xLAM ETC (81.29% versus 82.12%). This single-seed comparison supports the mixture choice, consistent with Hammer [13], without establishing an optimal mixing ratio. Complete checkpoint results for the positive-only control appear in Table 6 in Appendix B.

Preparation parses JSON fields of both datasets, converts Python-like parameter types to JSON Schema, and validates queries, unique tool names, and reference calls. We remove 260 positive rows and 20 negative rows whose offered tool lists contain repeated function names, ensuring that each function name uniquely identifies one offered tool within an example. We also remove 1,446 negative rows with no offered tools: our negative task is to abstain *despite* available tools, while the intended no-tools mode remains ordinary instruction following. No other source rows are removed. Table 1 gives the resulting counts. The mixture concatenates and shuffles the retained sources without oversampling.

Table 1: Prepared source splits. The mixture contains 65,774 retained rows; 1,726 of 67,500 source rows are removed.

| Source | Original | Removed | Train | Test |
|---|---|---|---|---|
| xLAM calls | 60,000 | 260 | 56,753 | 2,987 |
| Hammer irrelevant-tool examples | 7,500 | 1,466 | 5,732 | 302 |
| Prepared mixture | 67,500 | 1,726 | 62,485 | 3,289 |

After filtering, we first split the positive dataset into 95% training and 5% testing, keeping queries with identical text after whitespace normalization together. Negative examples with matching queries are assigned to the same split as their positive counterparts. We then assign the remaining negative query groups, aiming for 5% of all retained negative rows in testing, including those already assigned through matching queries. Keeping query groups together takes priority over reaching this target exactly. Finally, we combine the positive and negative examples within each split and shuffle them; preparation uses seed 42. This prevents exact-query overlap between the prepared training and test sets, but does not prevent paraphrase or tool-family overlap. At training time, both SFT and GRPO scripts reserve a further 5% of their length-filtered training rows for validation using a random row-based split with seed 42. The scripts support initial and periodic validation, but we disabled both in all reported training runs. The validation split was still created: these rows were excluded from training and were not evaluated during the runs. The prepared test set remains separate from this runtime split.

Complete SFT sequences above 896 tokens are dropped, retaining 57,088 mixed examples and removing 5,397. GRPO uses a 640-token prompt cap, retaining 48,791 and removing 13,694. Neither path truncates training inputs. Consequently, the stages share a source pool but have different eligible distributions.

### 3.2 Model and timing

All allocation runs use Qwen2.5-0.5B-Instruct [22], bfloat16 computation, low-rank adaptation (LoRA) [6] rank 8, $\alpha = 16$, zero dropout, and all linear layers as targets. There are 4,399,104 trainable parameters (0.8826% of the adapted model). Trainable LoRA parameters are float32. Every run uses seed 42. Both methods use global gradient clipping at 1.0 and AdamW [17] defaults $\beta = (0.9, 0.999)$, $\epsilon = 10^{-8}$, and weight decay 0.01.

Measured time starts immediately before each training loop and includes rollouts, updates, and periodic saving. It excludes startup, model loading, preprocessing, and external evaluation. Hybrid totals add the original SFT-prefix time, even though experiments reuse the checkpoint. The comparison is therefore training-loop wall time. Paper timings and checkpoint eligibility use the reported training times at their recorded precision.

### 3.3 Supervised fine-tuning

SFT uses the tokenizer's native tool-aware chat template and minimizes cross-entropy over assistant tokens, masking all other tokens and dividing the loss by the number of supervised tokens. No-call targets are assistant messages containing `[]`. We use batch size four, 200 warmup updates, and cosine decay from $5 \times 10^{-5}$ to $5 \times 10^{-6}$ over 19,600 planned updates. Checkpoints every 2,800 updates provide six GRPO initializations after 0.4, 0.9, 1.3, 1.8, 2.2, and 2.7 reported hours of SFT.

### 3.4 GRPO algorithm and hyperparameters

We use GRPO with token-level loss normalization, matching the reduction used in DAPO. Unlike the original GRPO objective, which averages token losses within each rollout before averaging across rollouts, we average over all generated tokens in the sampled group. We retain KL regularization and symmetric clipping, without DAPO's dynamic sampling. For each prompt $x$, GRPO [26] samples $G$ completions from the old policy. With rewards $R_i$, their mean $\bar{R} = G^{-1} \sum_j R_j$, and population variance $\sigma_R^2 = G^{-1} \sum_j (R_j - \bar{R})^2$, each completion receives advantage $A_i = \frac{R_i - \bar{R}}{\sigma_R + \delta}$ where $\delta$ stabilizes normalization. Let $m_{i,t}$ mask generated tokens, including the end-of-sequence token but excluding padding, and $N = \sum_{i,t} m_{i,t}$. For $a \in \{\theta, \text{old}, \text{ref}\}$, define token log probabilities $\ell_{i,t}^a = \log \pi_a(y_{i,t} \mid x, y_{i,<t})$. We compute the token probability ratio as $r_{i,t}(\theta) = \exp\left(\ell_{i,t}^\theta - \ell_{i,t}^{\text{old}}\right)$. Each inner update minimizes

$$\mathcal{L}_{\text{GRPO}} = -\frac{1}{N} \sum_{i,t} m_{i,t} \min\left(r_{i,t}(\theta) A_i, \text{clip}(r_{i,t}(\theta), 1-\epsilon, 1+\epsilon) A_i\right) + \beta \widehat{D}_{\text{KL}}(\theta), \tag{1}$$

where the sampled Kullback–Leibler (KL) penalty [9, 26] is

$$\widehat{D}_{\text{KL}}(\theta) = \frac{1}{N} \sum_{i,t} m_{i,t} \left[\text{expm1}(d_{i,t}) - d_{i,t}\right], \qquad d_{i,t} = \ell_{i,t}^{\text{ref}} - \ell_{i,t}^\theta.$$

Here $\epsilon$ controls clipping and $\beta$ weights the KL penalty. We evaluate $\text{expm1}(d) = e^d - 1$ for numerical stability. The loss averages over generated tokens across the group. Old-policy probabilities are fixed for that group; the reference remains frozen at each branch's initialization.

We use a deterministic binary reward function. For a completion $y$ and reference call multiset $C^\star$, let $C(y)$ be the normalized predicted call multiset and $b(y)$ indicate accepted abstention. The reward is

$$R(y, C^\star) = \begin{cases} 1, & C^\star \neq \varnothing \text{ and } C(y) = C^\star, \\ 1, & C^\star = \varnothing \text{ and } b(y) = 1, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

The local parser accepts Qwen `<tool_call>` blocks or bare JSON calls/lists, optionally in a complete JSON fence. It normalizes a nested `function` envelope and JSON-string arguments. Exact matching preserves function names, arguments, JSON types, and duplicate-call multiplicity, while ignoring call order. Accepted abstention includes `[]` and ordinary non-call text, but excludes parsed calls, attempted tool-call tags, and other JSON-like call attempts. This differs from requiring literal `[]` and from BFCL's parser. No learned reward model or partial credit is used.

All reported GRPO runs use one prompt per outer update with $G = 4$, $\epsilon = 0.2$, $\beta = 0.02$, and $\delta = 10^{-8}$. Sampling allows at most 256 new tokens, with temperature 1, top-$p$ 1, top-$k$ disabled, and repetition penalty 1. Four inner AdamW updates reuse each sampled group at learning rate $10^{-6}$; old-policy and reference log probabilities are computed once per group. Log probabilities, ratios, and KL calculations use float32.

### 3.5 Evaluation

Our local xLAM evaluator reports exact tool-call accuracy (ETC) and function-name accuracy (FNA). ETC requires all function names, argument keys, JSON types and values, and call multiplicities to match; call and object-key order are ignored, while array order is preserved. FNA checks only function names and their multiplicities, ignoring arguments. Both are whole-example accuracies without partial credit, measured on the same 2,287 positive examples retained from

2,987 held-out rows by the 640-token prompt cap; 700 (23.43%) are excluded. This holdout measures call construction but cannot measure abstention; the prepared mixed holdout remains unevaluated.

Selected Berkeley Function Calling Leaderboard (BFCL-v3) subsets [19], evaluated with EvalScope [29], test external call construction and appropriate tool use. BFCL-v3 Single-turn FC (agg) measures single-turn function-calling accuracy using abstract syntax tree checks of function names and arguments across eight subsets containing 2,351 examples in total. BFCL-v3 Irrelevance (agg) measures correct abstention when no offered tool is suitable, combining `irrelevance` and `live_irrelevance`, which contain 1,122 examples in total. We compute these two aggregates separately, weighting each subset's accuracy by its number of examples: $100\frac{\sum_s n_s a_s}{\sum_s n_s}$. Here, $s$ indexes the subsets included in the aggregate, $n_s$ is the number of examples in subset $s$, and $a_s \in [0, 1]$ is the fraction of those examples answered correctly. The factor of 100 converts the result to a percentage. These are study-specific summaries rather than official BFCL leaderboard aggregates. Throughout the paper, the suffix (agg) identifies these two aggregates, including shortened labels in tables and figures. Relevance measures appropriate invocation when a suitable tool is offered. Its 18 examples yield identical scores for seven of eight adapted endpoints, so we report it only in the complete appendix results (Table 5).

We use lm-evaluation-harness [2] to probe selected capabilities without tool schemas: Massive Multitask Language Understanding (MMLU) measures knowledge and problem solving across 57 subjects using subject-size-weighted accuracy [4]; MATH-500 measures mathematical reasoning using answer-equivalence accuracy [5, 12]; and Instruction-Following Evaluation (IFEval) measures compliance with verifiable instructions using strict prompt-level accuracy, requiring every constraint in a prompt to pass [39]. These diagnostics assess knowledge, reasoning, and instruction following beyond tool calling rather than unrestricted conversational quality. All reported metrics are percentages, with higher values better.

## 4 Results

At the last evaluated checkpoints within the 3.3-hour envelope, mixed-data SFT reaches 82.82% xLAM ETC at 3.1 hours, compared with 69.13% for GRPO alone at 3.3 hours and 63.84% for the starting model (Table 2). The strongest hybrid endpoint reaches 81.90%. SFT therefore leads local exact-call specialization under the primary reporting rule. Figure 1 shows how tool-calling and general-capability metrics evolve over the sampled checkpoints.

The SFT trajectory is not monotonic. Between 0.9 and 1.3 reported hours, xLAM ETC falls from 77.61% to 71.49% and FNA from 97.46% to 86.88%, while BFCL-v3 Irrelevance (agg) rises from 49.46% to 88.59%. These changes are consistent with a temporary shift toward abstention, but aggregate scores do not establish its cause. Later SFT checkpoints recover call-construction accuracy, so SFT-prefix duration alone does not imply steadily improving specialization.

GRPO alone has the lowest xLAM ETC, xLAM FNA, and BFCL-v3 Single-turn FC (agg) among the adapted endpoints; its 50.24% BFCL-v3 Single-turn FC (agg) remains close to the starting model's 50.28%. Its leading BFCL-v3 Irrelevance (agg) score (74.60%) therefore coexists with weak call construction. One possible explanation is that it remains less specialized, preserving more of the starting model's behavior rather than learning a better overall tool-use policy. GRPO also preserves MMLU better than the other adapted models, so its weakness is specific to call construction rather than every measured capability.

Training diagnostics suggest that sparse relative task feedback may contribute to GRPO's slow adaptation. Recalling Section 3.4, $A_i = \frac{R_i - \bar{R}}{\sigma_R + \delta}$: all-correct and all-incorrect groups have constant rewards and therefore zero task advantages. KL regularization, AdamW momentum, and weight decay can still change parameters. Across the selected primary endpoints, 74.87% of GRPO-only groups and 81.65–84.09% of groups in the six hybrid branches have constant rewards; 66.41–75.91% of hybrid groups are all correct. All-correct groups are already frequent early in the hybrid runs. These fractions use nonoverlapping ten-prompt TensorBoard intervals and count groups rather than completions or inner updates; Appendix A reports the full summaries.

These frequent constant-reward groups raise the question of whether larger groups could provide more useful task feedback. With binary rewards, a group provides relative task feedback only when it contains both correct and incorrect completions. For a given prompt, let $p$ be the probability that one sampled completion is correct and $G$ the number of completions in the group. Assuming independent samples, the probability of mixed outcomes is $1 - p^G - (1-p)^G$ [18]: $p^G$ is the probability that all completions are correct, and $(1-p)^G$ is the probability that all are incorrect. Here, $p$ describes one prompt, not the overall accuracy across prompts, which may differ in difficulty. Larger groups increase the chance of mixed rewards when $0 < p < 1$, but also increase rollout cost. Whether this trade-off improves learning within the same training-time budget remains untested here.

Table 2: Primary comparison using the last evaluated checkpoint at or below 3.3 hours. Hybrid times include SFT initialization. Scores are percentages (↑: higher is better); bold marks the best score in each metric column, including ties and the base model.

| Recipe | Training time (h) | | | xLAM | | BFCL-v3 | | General capabilities | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | SFT | GRPO | Total | ETC ↑ | FNA ↑ | Single-turn FC (agg) ↑ | Irrelevance (agg) ↑ | MMLU ↑ | MATH-500 ↑ | IFEval ↑ |
| Base | 0.0 | 0.0 | 0.0 | 63.84 | 90.03 | 50.28 | 66.93 | **42.04** | 15.80 | 26.80 |
| SFT | 3.1 | 0.0 | 3.1 | **82.82** | 97.81 | 56.78 | 67.65 | 38.19 | 17.20 | 30.13 |
| GRPO | 0.0 | 3.3 | 3.3 | 69.13 | 92.26 | 50.24 | **74.60** | 41.81 | 15.40 | 25.51 |
| SFT→GRPO | 0.4 | 2.6 | 3.0 | 76.17 | 96.20 | 55.17 | 59.98 | 39.48 | 17.60 | 31.98 |
| SFT→GRPO | 0.9 | 2.2 | 3.1 | 79.49 | 97.38 | **57.59** | 54.90 | 38.74 | 17.20 | **33.09** |
| SFT→GRPO | 1.3 | 1.8 | 3.1 | 79.49 | 96.90 | 56.23 | 66.31 | 37.96 | **18.80** | 29.94 |
| SFT→GRPO | 1.8 | 1.5 | 3.3 | 80.72 | 97.64 | 56.53 | 63.81 | 38.15 | 17.40 | 30.13 |
| SFT→GRPO | 2.2 | 1.1 | 3.3 | 80.94 | 97.11 | 57.46 | 69.69 | 37.95 | 17.80 | 28.10 |
| SFT→GRPO | 2.7 | 0.4 | 3.1 | 81.90 | **98.03** | 57.42 | 54.55 | 38.46 | 15.60 | 29.02 |

Adding GRPO after SFT changes the trade-off. The 0.9-hour SFT prefix produces the highest endpoint BFCL-v3 Single-turn FC (agg) (57.59%) and IFEval (33.09%) at 3.1 total hours, but BFCL-v3 Irrelevance (agg) falls to 54.90%, versus SFT's 67.65%. A 2.2-hour SFT prefix reaches nearly the same BFCL-v3 Single-turn FC (agg) (57.46%) with 69.69% BFCL-v3 Irrelevance (agg). Thus 0.9 hours is a promising handoff for these two metrics, not a universal best allocation. Under this budget, SFT is a strong first baseline; a GRPO continuation should be selected against the intended call-construction and abstention criteria rather than assumed to improve both.

This ranking differs from ToolRL and Tool-N1, which find that RL directly from instruction-tuned models can outperform both SFT and SFT-to-RL under their recipes [20, 36]. Those comparisons do not impose our short training-time envelope; data, rewards, and optimization also differ. Tool-N1 additionally reports limited RL gains at 0.5B and 1.5B, making model scale a relevant qualification. Our local SFT advantage agrees qualitatively with Laskar et al.'s predominantly SFT-favoring in-distribution results [10]. Together, these findings favor evaluating the recipe under the actual resource budget rather than extrapolating an RL advantage across settings.

The general-capability probes show selective changes. Relative to the starting model, SFT raises MATH-500 from 15.80% to 17.20% and IFEval from 26.80% to 30.13%, while MMLU falls from 42.04% to 38.19%. Five of six hybrid endpoints also improve both MATH-500 and IFEval; the 2.7-hour prefix is the exception on MATH-500 (15.60%). GRPO alone scores below the starting model on all three probes, although its 41.81% MMLU is the best adapted score. Tool-specialization gains can therefore accompany selected non-tool improvements, but neither general-capability preservation nor a uniform loss follows from these results. These probes complement the tool-use transfer comparisons in the closest studies rather than measuring the same form of generalization.

Table 3 summarizes the observed recipes for different practitioner priorities, including earlier evaluated checkpoints alongside the primary endpoints. Approximate SFT/GRPO shares describe how training time was divided between stages; early stopping is a separate choice. These descriptive recommendations apply to the measured setting and require validation for other models, datasets, or hardware.

Table 3: Observed recipes for practitioner priorities. For hybrid recipes, percentages indicate the share of training time allocated to SFT and GRPO. BFCL FC (agg) and Irrelevance (agg) denote the study-specific BFCL-v3 aggregates. These proportions can be applied to other time budgets, but comparable performance at those budgets has not been established.

| Practitioner priority | Recipe (time allocation) | Pros | Cons |
|---|---|---|---|
| Keep training simple with strong tool-calling results. | SFT only, 3.1 h | Improves all four tool-calling scores over the base: ETC 82.82, FNA 97.81, BFCL FC (agg) 56.78, and Irrelevance (agg) 67.65. Highest observed ETC; one training stage. | Some hybrids achieve slightly higher FNA or BFCL FC (agg). MMLU declines, although MATH-500 and IFEval improve. |

Table 3: Observed recipes for practitioner priorities (continued).

| Practitioner priority | Recipe (time allocation) | Pros | Cons |
|---|---|---|---|
| Reduce training time while keeping performance close to the full SFT run. | SFT only, 2.2 h | Saves 0.9 h. All four tool-calling scores remain within 1.5 points of final SFT: ETC 81.33, FNA 97.68, BFCL FC (agg) 56.19, Irrelevance (agg) 66.40. | Gives up some performance on every tool-calling metric relative to final SFT. Irrelevance (agg) is slightly below the base model's 66.93. MMLU is also lower than the base. |
| Maximize selected tool-calling scores: prioritize correct function names. | 2.7 h SFT + 0.4 h GRPO<br>$\sim 87\%$ SFT + $\sim 13\%$ GRPO | Highest observed FNA: 98.03. BFCL FC (agg) reaches 57.42, improving on SFT. ETC remains close to SFT at 81.90, only 0.92 points lower. | Irrelevance (agg) falls substantially to 54.55, a 13.10-point reduction. The FNA advantage is small. MATH-500 is slightly below the base model. |
| Keep general scores close to the base while gaining some tool-calling ability. | GRPO only, 1.9 h | MMLU is 41.80 versus the base's 42.04; MATH-500 and IFEval improve. All four tool-calling scores exceed the base. | Tool-calling gains are limited: ETC 69.70 and BFCL FC (agg) 50.70, substantially below stronger SFT and hybrid recipes. General scores are not strictly unchanged. |
| Correctly avoid unsuitable tools while obtaining higher BFCL function call accuracy. | 2.2 h SFT + 1.1 h GRPO<br>$\sim 70\%$ SFT + $\sim 30\%$ GRPO | BFCL FC (agg) 57.46 and Irrelevance (agg) 69.69 both exceed final SFT. BFCL FC (agg) is only 0.13 points below the highest observed score. | Two training stages. ETC 80.94, FNA 97.11, and MMLU are below final SFT. |

## 5 Limitations

Our experiments focus on one small instruction-tuned model, one consumer GPU, and a short training budget, using a single random seed per trajectory. The findings characterize the SFT–GRPO trade-offs within this setting. Repeating the comparison across seeds, model families, and budgets would assess the robustness of the observed rankings, particularly where score differences are small. Stage-specific length limits also mean that the comparison reflects the feasible training recipes, including their eligible data, rather than the training objective alone.

The reward-group diagnostics identify frequent groups without relative task feedback in the studied GRPO recipe, suggesting one explanation for its slower improvement. Targeted comparisons of group sizes and reward designs would help separate their effects on learning. Keeping the training budget fixed would test whether more informative groups improve learning enough to offset any additional rollout cost. Earlier binary/partial pilots differ in mixture, group size, sampling, or precision and do not constitute a controlled reward ablation.

Evaluation emphasizes single-turn call construction and abstention when available tools are unsuitable, with abstention assessed through BFCL because the local holdout contains only positive calls. Evaluating the prepared mixed holdout and multi-turn tool-use tasks would broaden the behavioral assessment. MMLU, MATH-500, and IFEval provide complementary capability probes. Dedicated conversational evaluation would further assess ordinary responses and unintended tool calls when no tools are offered.

## 6 Conclusion

We studied how to allocate a short training budget between SFT and GRPO when specializing a small language model for tool calling on one consumer GPU. Within the measured 3.3-hour envelope, SFT achieves the highest primary-endpoint xLAM ETC and provides a simple route to strong tool-calling performance. Selected hybrids improve function-name accuracy or external BFCL call accuracy, with different abstention and general-capability trade-offs. GRPO alone improves call construction less, despite its high BFCL-v3 Irrelevance (agg) score, and training logs show that most sampled groups supply no relative task-reward signal. These findings support starting with SFT and judging a GRPO continuation against the intended evaluation priorities. The released mixed dataset, checkpoints, and Post-Training Lab implementation support reproduction and further study of these budget-dependent choices.

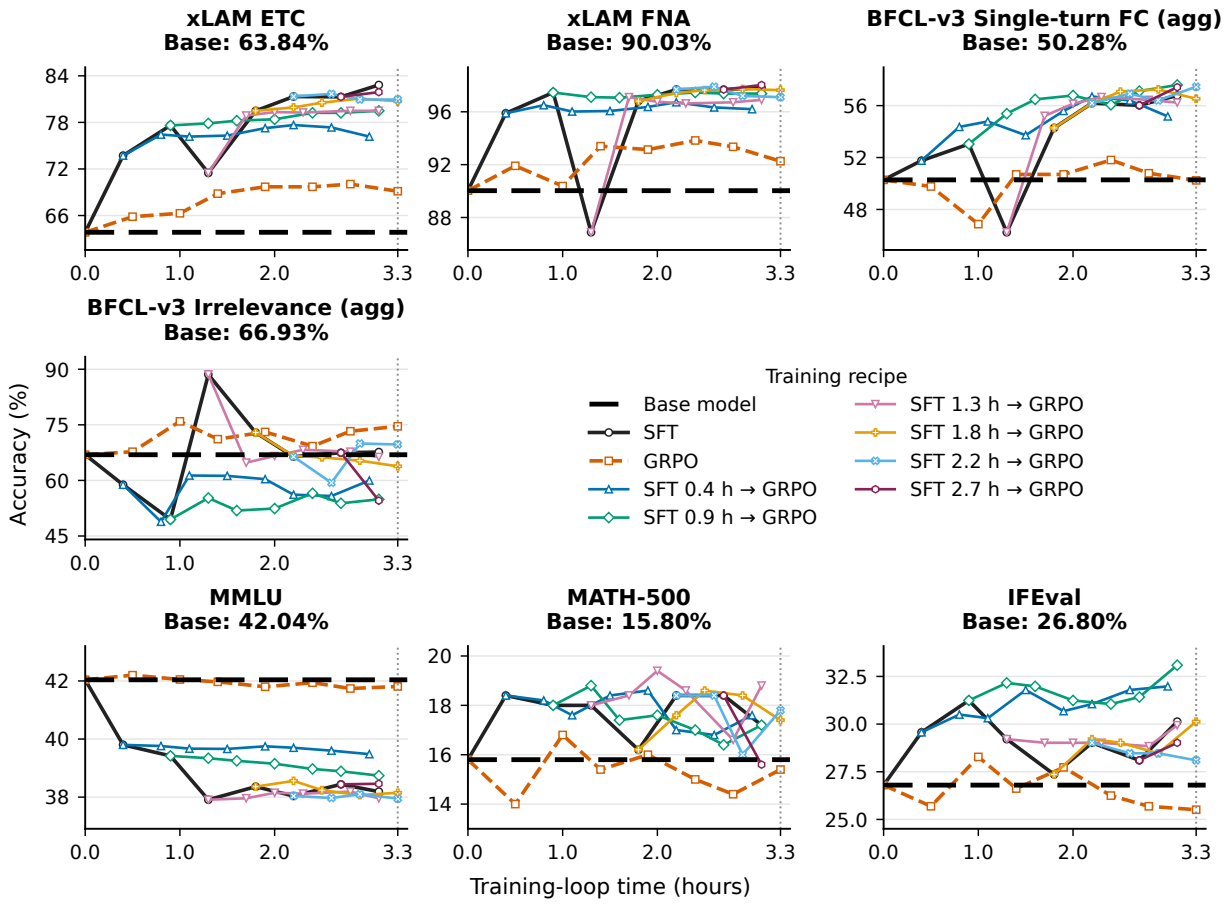



Figure 1: Tool-calling and general-capability scores over training time for SFT, GRPO, and six SFT-to-GRPO allocations. Hybrid times include SFT initialization. Dashed horizontal lines show base-model scores; dotted vertical lines mark the 3.3-hour cutoff. Lines connect evaluated checkpoints from single-seed runs; axis ranges vary by metric. The suffix (agg) denotes the study-specific BFCL-v3 aggregates.

## References


[1] Y. Fu, T. Chen, J. Chai, X. Wang, S. Tu, G. Yin, W. Lin, Q. Zhang, Y. Zhu, and D. Zhao. SRFT: A Single-Stage Method with Supervised and Reinforcement Fine-Tuning for Reasoning. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/a79699db176ed0efc04a9da171e52112-Abstract-Conference.html.

[2] L. Gao, J. Tow, B. Abbasi, S. Biderman, S. Black, A. DiPofi, C. Foster, L. Golding, J. Hsu, A. Le Noac'h, H. Li, K. McDonell, N. Muennighoff, C. Ociepa, J. Phang, L. Reynolds, H. Schoelkopf, A. Skowron, L. Sutawika, E. Tang, A. Thite, B. Wang, K. Wang, and A. Zou. The language model evaluation harness, 07 2024. URL https://zenodo.org/records/12608602.

[3] J. Geiping and T. Goldstein. Cramming: Training a Language Model on a Single GPU in One Day. In *Proceedings of ICML*, 2023. URL https://proceedings.mlr.press/v202/geiping23a.html.

[4] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt. Measuring Massive Multitask Language Understanding, 2021. URL https://arxiv.org/abs/2009.03300. arXiv:2009.03300 [cs.CY].

[5] D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, and J. Steinhardt. Measuring Mathematical Problem Solving With the MATH Dataset, 2021. URL https://arxiv.org/abs/2103.03874. arXiv:2103.03874 [cs.LG].

[6] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. LoRA: Low-Rank Adaptation of Large Language Models, 2021. URL https://arxiv.org/abs/2106.09685. arXiv:2106.09685 [cs.CL].

[7] J. Hu, X. Wu, W. Shen, J. K. Liu, Z. Zhu, W. Wang, S. Jiang, H. Wang, H. Chen, B. Chen, W. Fang, Xianyu, Y. Cao, H. Xu, and Y. Liu. OpenRLHF: An easy-to-use, scalable and high-performance RLHF framework, 2025. URL https://arxiv.org/abs/2405.11143v6.

[8] F. Kang, M. Kuchnik, K. Padthe, M. Vlastelica, R. Jia, C.-J. Wu, and N. Ardalani. Quagmires in SFT-RL Post-Training: When High SFT Scores Mislead and What to Use Instead. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/5d6ae8ba43ecb378030753c4408ef9bd-Abstract-Conference.html.

[9] S. Kullback and R. A. Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[10] M. T. R. Laskar, X.-Y. Fu, and S. B. TN. SFT or RL for Tool-Calling Agents? A Controlled Study Across Data, Method, and Scale, 2026. URL https://arxiv.org/abs/2609.17848. arXiv:2609.17848 [cs.CL].

[11] X. Li, G. Huzhang, S. Shen, Q.-G. Chen, Z. Xu, W. Luo, K. Zhang, and J. Zhang. Getting Your LLMs Ready for Reinforcement Learning with Lightweight SFT. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/bebed4d2aaa8a0ebfcc9fd55ce811e83-Abstract-Conference.html.

[12] H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe. Let's verify step by step, 2023. URL https://arxiv.org/abs/2305.20050.

[13] Q. Lin, M. Wen, Q. Peng, G. Nie, J. Liao, J. Wang, X. Mo, J. Zhou, C. Cheng, Y. Zhao, J. Wang, and W. Zhang. Hammer: Robust Function-Calling for On-Device Language Models via Function Masking, 2024. URL https://arxiv.org/abs/2410.04587. arXiv:2410.04587 [cs.LG].

[14] T. Liu, C. Qian, M. Cief, Y. He, D. Dan, N. Aletras, and G. Kazai. On Effectiveness and Efficiency of Agentic Tool-calling and RL Training, 2026. URL https://arxiv.org/abs/2606.00135.

[15] W. Liu, X. Huang, X. Zeng, X. Hao, S. Yu, D. Li, S. Wang, W. Gan, Z. Liu, Y. Yu, Z. Wang, Y. Wang, W. Ning, Y. Hou, B. Wang, C. Wu, X. Wang, Y. Liu, Y. Wang, D. Tang, D. Tu, L. Shang, X. Jiang, R. Tang, D. Lian, Q. Liu, and E. Chen. ToolACE: Winning the Points of LLM Function Calling, 2025. URL https://arxiv.org/abs/2409.00920. arXiv:2409.00920 [cs.LG].

[16] Z. Liu, T. Hoang, J. Zhang, M. Zhu, T. Lan, S. Kokane, J. Tan, W. Yao, Z. Liu, Y. Feng, R. Murthy, L. Yang, S. Savarese, J. C. Niebles, H. Wang, S. Heinecke, and C. Xiong. APIGen: Automated Pipeline for Generating Verifiable and Diverse Function-Calling Datasets, 2024. URL https://arxiv.org/abs/2406.18518. arXiv:2406.18518 [cs.CL].

[17] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[18] W. Nie, J. Wu, J. Liu, Z. Li, Z. Lin, Zhang Zijian, Y. Fan, H. Zheng, and J.-S. R. Jang. Gradient Starvation in Binary-Reward GRPO: Why Group-Mean Centering Fails and Why the Simplest Fix Works, 2026. URL https://arxiv.org/abs/2605.07689. arXiv:2605.07689 [cs.LG].

[19] S. G. Patil, H. Mao, F. Yan, C. C.-J. Ji, V. Suresh, I. Stoica, and J. E. Gonzalez. The Berkeley Function Calling Leaderboard (BFCL): From Tool Use to Agentic Evaluation of Large Language Models. In *Proceedings of ICML*, 2025. URL https://proceedings.mlr.press/v267/patil25a.html.

[20] C. Qian, E. C. Acikgoz, Q. He, H. Wang, X. Chen, D. Hakkani-Tür, G. Tur, and H. Ji. ToolRL: Reward is All Tool Learning Needs, 2025. URL https://arxiv.org/abs/2504.13958. arXiv:2504.13958 [cs.LG].

[21] Y. Qin, S. Liang, Y. Ye, K. Zhu, L. Yan, Y. Lu, Y. Lin, X. Cong, X. Tang, B. Qian, S. Zhao, L. Hong, R. Tian, R. Xie, J. Zhou, M. Gerstein, D. Li, Z. Liu, and M. Sun. ToolLLM: Facilitating Large Language Models to Master 16000+ Real-world APIs, 2023. URL https://arxiv.org/abs/2307.16789. arXiv:2307.16789 [cs.AI].

[22] Qwen, A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei, H. Lin, J. Yang, J. Tu, J. Zhang, J. Yang, J. Yang, J. Zhou, J. Lin, K. Dang, K. Lu, K. Bao, K. Yang, L. Yu, M. Li, M. Xue, P. Zhang, Q. Zhu, R. Men, R. Lin, T. Li, T. Tang, T. Xia, X. Ren, X. Ren, Y. Fan, Y. Su, Y. Zhang, Y. Wan, Y. Liu, Z. Cui, Z. Zhang, and Z. Qiu. Qwen2.5 Technical Report, 2025. URL https://arxiv.org/abs/2412.15115. arXiv:2412.15115 [cs.CL].

[23] M. Raghavendra, J. Kang, and A. Ritter. Balancing the Budget: Understanding Trade-offs Between Supervised and Preference-Based Finetuning. In *Proceedings of ACL*, 2025. doi: 10.18653/v1/2025.acl-long.1248. URL https://aclanthology.org/2025.acl-long.1248/.

[24] H. Ross, A. S. Mahabaleshwarkar, and Y. Suhara. When2Call: When (not) to Call Tools, 2025. URL https://arxiv.org/abs/2504.18851. arXiv:2504.18851.

[25] T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, L. Zettlemoyer, N. Cancedda, and T. Scialom. Toolformer: Language Models Can Teach Themselves to Use Tools, 2023. URL https://arxiv.org/abs/2302.04761. arXiv:2302.04761 [cs.CL].

[26] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. K. Li, Y. Wu, and D. Guo. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models, 2024. URL https://arxiv.org/abs/2402.03300. arXiv:2402.03300 [cs.CL].

[27] G. Sheng, C. Zhang, Z. Ye, X. Wu, W. Zhang, R. Zhang, Y. Peng, H. Lin, and C. Wu. HybridFlow: A flexible and efficient RLHF framework, 2024. URL https://arxiv.org/abs/2409.19256.

[28] M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, and B. Catanzaro. Megatron-LM: Training multi-billion parameter language models using model parallelism, 2019. URL https://arxiv.org/abs/1909.08053.

[29] M. Team. EvalScope: Evaluation framework for large models, 2024. URL https://github.com/modelscope/evalscope.

[30] D. Vattikonda, S. Ravichandran, E. Penaloza, H. Nekoei, M. Thakkar, T. Le Sellier de Chezelles, N. Gontier, M. Muñoz-Mármol, S. Omidi Shayegan, S. Raimondo, X. Liu, A. Drouin, L. Charlin, A. Piché, A. Lacoste, and M. Caccia. How to Train Your LLM Web Agent: A Statistical Diagnosis, 2025. URL https://arxiv.org/abs/2507.04103.

[31] J. Wang, A. Verma, X. Lin, Z. Liu, N. F. Chen, D. Rus, and B. K. H. Low. Scaling Near-Optimal SFT-RL Annotation Budget Allocation from Small to Large LLMs, 2026. URL https://arxiv.org/abs/2609.01573. arXiv:2609.01573 [cs.CL].

[32] Q. Yu, Z. Zhang, R. Zhu, Y. Yuan, X. Zuo, Y. Yue, W. Dai, T. Fan, G. Liu, L. Liu, X. Liu, H. Lin, Z. Lin, B. Ma, G. Sheng, Y. Tong, C. Zhang, M. Zhang, W. Zhang, H. Zhu, J. Zhu, J. Chen, J. Chen, C. Wang, H. Yu, Y. Song, X. Wei, H. Zhou, J. Liu, W.-Y. Ma, Y.-Q. Zhang, L. Yan, M. Qiao, Y. Wu, and M. Wang. DAPO: An Open-Source LLM Reinforcement Learning System at Scale, 2025. URL https://arxiv.org/abs/2503.14476. arXiv:2503.14476 [cs.LG].

[33] Y. Zhan, S. Liu, Q. Liu, Y. Shi, S. Xu, W. Hou, X. Xu, Z. Li, W. Pan, and J. Yan. SLCA-GRPO: Resolving Cross-Segment Credit Misattribution in Tool-Calling RL, 2026. URL https://arxiv.org/abs/2609.29050. arXiv:2609.29050.

[34] J. Zhang, T. Lan, M. Zhu, Z. Liu, T. Hoang, S. Kokane, W. Yao, J. Tan, A. Prabhakar, H. Chen, Z. Liu, Y. Feng, T. Awalgaonkar, R. Murthy, E. Hu, Z. Chen, R. Xu, J. C. Niebles, S. Heinecke, H. Wang, S. Savarese, and C. Xiong. xLAM: A Family of Large Action Models to Empower AI Agent Systems, 2024. URL https://arxiv.org/abs/2409.03215. arXiv:2409.03215 [cs.CL].

[35] J. Zhang, T. Hoang, M. Zhu, Z. Liu, S. Wang, T. Awalgaonkar, A. Prabhakar, H. Chen, W. Yao, Z. Liu, J. Tan, J. C. Niebles, S. Heinecke, H. Wang, S. Savarese, and C. Xiong. ActionStudio: A Lightweight Framework for Data and Training of Large Action Models, 2025. URL https://arxiv.org/abs/2503.22673. arXiv:2503.22673 [cs.AI].

[36] S. Zhang, Y. Dong, J. Zhang, J. Kautz, B. Catanzaro, A. Tao, Q. Wu, Z. Yu, and G. Liu. Nemotron-Research-Tool-N1: Exploring Tool-Using Language Models with Reinforced Reasoning, 2025. URL https://arxiv.org/abs/2505.00024. arXiv:2505.00024 [cs.CL].

[37] Y. Zheng, R. Zhang, J. Zhang, Y. Ye, Z. Luo, Z. Feng, and Y. Ma. LlamaFactory: Unified Efficient Fine-Tuning of 100+ Language Models, 2024. URL https://arxiv.org/abs/2403.13372. arXiv:2403.13372.

[38] H. Zhong, J. Zhai, L. Song, J. Bian, Q. Liu, and T. Tan. RC-GRPO: Reward-Conditioned Group Relative Policy Optimization for Multi-Turn Tool Calling Agents, 2026. URL https://arxiv.org/abs/2602.03025. arXiv:2602.03025 [cs.AI].

[39] J. Zhou, T. Lu, S. Mishra, S. Brahma, S. Basu, Y. Luan, D. Zhou, and L. Hou. Instruction-Following Evaluation for Large Language Models, 2023. URL https://arxiv.org/abs/2311.07911. arXiv:2311.07911 [cs.CL].

# A Reward-group diagnostics

Table 4: Training-group diagnostics. Rates are percentages of prompt groups. Each interval contains ten groups.

| SFT h | Outer step | All zero | All one | Mixed | Zero advantage |
|---|---|---|---|---|---|
| 0.0 | 4620 | 23.25 | 51.62 | 25.13 | 74.87 |
| 0.4 | 4620 | 15.24 | 66.41 | 18.35 | 81.65 |
| 0.9 | 3960 | 13.06 | 68.74 | 18.21 | 81.79 |
| 1.3 | 3300 | 10.97 | 71.03 | 18.00 | 82.00 |
| 1.8 | 2640 | 10.42 | 71.74 | 17.84 | 82.16 |
| 2.2 | 1980 | 8.94 | 75.00 | 16.06 | 83.94 |
| 2.7 | 660 | 8.18 | 75.91 | 15.91 | 84.09 |

The fractions are averaged over equal-sized, nonoverlapping ten-prompt intervals through each listed checkpoint. For binary rewards, all-zero, all-one, and mixed fractions sum to one; the zero-task-advantage fraction is the sum of the first two. Each outer step uses one prompt group and four sampled completions. All six hybrid branches have an evaluated continuation within the primary 3.3-hour cutoff. The branch initialized after 2.7 reported SFT hours has 660 prompt groups through its first continuation at 3.1 reported total hours.

# B Complete experiment table

Table 5: Complete checkpoint results for the mixed-data SFT, GRPO, and SFT-to-GRPO recipes. Hybrid times include SFT initialization; rows with zero GRPO time repeat the corresponding SFT starting checkpoint. The positive-only control is reported separately in Table 6. Scores are percentages (↑: higher is better); bold marks the highest score in each metric column within this table, including ties and the base model.

| Recipe | Training time (h) | | | xLAM | | BFCL-v3 | | | General capabilities | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | SFT | GRPO | Total | ETC ↑ | FNA ↑ | Single-turn FC (agg) ↑ | Relevance ↑ | Irrelevance (agg) ↑ | MMLU ↑ | MATH-500 ↑ | IFEval ↑ |
| Base | 0.0 | 0.0 | 0.0 | 63.84 | 90.03 | 50.28 | 77.78 | 66.93 | 42.04 | 15.80 | 26.80 |
| SFT | 0.4 | 0.0 | 0.4 | 73.72 | 95.89 | 51.76 | 83.33 | 58.83 | 39.80 | 18.40 | 29.57 |
| SFT | 0.9 | 0.0 | 0.9 | 77.61 | 97.46 | 53.04 | 83.33 | 49.46 | 39.42 | 18.00 | 31.24 |
| SFT | 1.3 | 0.0 | 1.3 | 71.49 | 86.88 | 46.24 | 55.56 | **88.59** | 37.91 | 18.00 | 29.21 |
| SFT | 1.8 | 0.0 | 1.8 | 79.49 | 96.81 | 54.28 | 83.33 | 72.91 | 38.36 | 16.20 | 27.36 |
| SFT | 2.2 | 0.0 | 2.2 | 81.33 | 97.68 | 56.19 | 83.33 | 66.40 | 38.04 | 18.40 | 29.02 |
| SFT | 2.7 | 0.0 | 2.7 | 81.29 | 97.68 | 56.02 | 83.33 | 67.47 | 38.44 | 18.40 | 28.10 |
| SFT | 3.1 | 0.0 | 3.1 | **82.82** | 97.81 | 56.78 | 83.33 | 67.65 | 38.19 | 17.20 | 30.13 |
| GRPO | 0.0 | 0.5 | 0.5 | 65.85 | 91.91 | 49.77 | 77.78 | 67.74 | **42.20** | 14.00 | 25.69 |
| GRPO | 0.0 | 1.0 | 1.0 | 66.29 | 90.38 | 46.87 | 77.78 | 75.94 | 42.05 | 16.80 | 28.28 |
| GRPO | 0.0 | 1.4 | 1.4 | 68.82 | 93.40 | 50.70 | 77.78 | 71.13 | 41.97 | 15.40 | 26.62 |
| GRPO | 0.0 | 1.9 | 1.9 | 69.70 | 93.14 | 50.70 | 77.78 | 73.08 | 41.80 | 16.00 | 27.73 |
| GRPO | 0.0 | 2.4 | 2.4 | 69.70 | 93.83 | 51.81 | 72.22 | 69.25 | 41.94 | 15.00 | 26.25 |
| GRPO | 0.0 | 2.8 | 2.8 | 70.05 | 93.35 | 50.79 | 77.78 | 73.27 | 41.74 | 14.40 | 25.69 |
| GRPO | 0.0 | 3.3 | 3.3 | 69.13 | 92.26 | 50.24 | 77.78 | 74.60 | 41.81 | 15.40 | 25.51 |
| SFT→GRPO | 0.4 | 0.0 | 0.4 | 73.72 | 95.89 | 51.76 | 83.33 | 58.83 | 39.80 | 18.40 | 29.57 |
| SFT→GRPO | 0.4 | 0.4 | 0.8 | 76.43 | 96.50 | 54.36 | **88.89** | 48.84 | 39.76 | 18.20 | 30.50 |
| SFT→GRPO | 0.4 | 0.7 | 1.1 | 76.17 | 96.02 | 54.78 | 83.33 | 61.32 | 39.67 | 17.60 | 30.31 |
| SFT→GRPO | 0.4 | 1.1 | 1.5 | 76.30 | 96.06 | 53.72 | 83.33 | 61.23 | 39.66 | 18.40 | 31.79 |
| SFT→GRPO | 0.4 | 1.5 | 1.9 | 77.26 | 96.37 | 55.59 | 83.33 | 60.34 | 39.75 | 18.60 | 30.68 |
| SFT→GRPO | 0.4 | 1.8 | 2.2 | 77.66 | 96.72 | 56.70 | 83.33 | 56.15 | 39.70 | 17.00 | 31.05 |
| SFT→GRPO | 0.4 | 2.2 | 2.6 | 77.35 | 96.33 | 56.40 | 83.33 | 55.79 | 39.60 | 16.80 | 31.79 |
| SFT→GRPO | 0.4 | 2.6 | 3.0 | 76.17 | 96.20 | 55.17 | 83.33 | 59.98 | 39.48 | 17.60 | 31.98 |

Table 5 (continued)

| Recipe | Training time (h) | | | xLAM | | BFCL-v3 | | | General capabilities | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | SFT | GRPO | Total | ETC ↑ | FNA ↑ | Single-turn FC (agg) ↑ | Relevance ↑ | Irrelevance (agg) ↑ | MMLU ↑ | MATH-500 ↑ | IFEval ↑ |
| SFT→GRPO | 0.9 | 0.0 | 0.9 | 77.61 | 97.46 | 53.04 | 83.33 | 49.46 | 39.42 | 18.00 | 31.24 |
| SFT→GRPO | 0.9 | 0.4 | 1.3 | 77.87 | 97.11 | 55.38 | 83.33 | 55.26 | 39.34 | 18.80 | 32.16 |
| SFT→GRPO | 0.9 | 0.7 | 1.6 | 78.22 | 97.07 | 56.48 | 83.33 | 51.87 | 39.25 | 17.40 | 31.98 |
| SFT→GRPO | 0.9 | 1.1 | 2.0 | 78.40 | 97.29 | 56.78 | 83.33 | 52.41 | 39.15 | 17.60 | 31.24 |
| SFT→GRPO | 0.9 | 1.5 | 2.4 | 79.23 | 97.46 | 56.06 | 83.33 | 56.51 | 38.97 | 17.00 | 31.05 |
| SFT→GRPO | 0.9 | 1.8 | 2.7 | 79.23 | 97.38 | 57.12 | 83.33 | 53.83 | 38.89 | 16.40 | 31.42 |
| SFT→GRPO | 0.9 | 2.2 | 3.1 | 79.49 | 97.38 | **57.59** | 83.33 | 54.90 | 38.74 | 17.20 | **33.09** |
| SFT→GRPO | 1.3 | 0.0 | 1.3 | 71.49 | 86.88 | 46.24 | 55.56 | **88.59** | 37.91 | 18.00 | 29.21 |
| SFT→GRPO | 1.3 | 0.4 | 1.7 | 78.88 | 97.11 | 55.21 | 83.33 | 64.79 | 37.96 | 18.40 | 29.02 |
| SFT→GRPO | 1.3 | 0.7 | 2.0 | 79.32 | 96.76 | 56.14 | 83.33 | 66.58 | 38.15 | **19.40** | 29.02 |
| SFT→GRPO | 1.3 | 1.0 | 2.3 | 79.27 | 96.63 | 56.66 | 83.33 | 68.27 | 38.12 | 18.60 | 29.02 |
| SFT→GRPO | 1.3 | 1.5 | 2.8 | 79.49 | 96.72 | 56.44 | 83.33 | 67.74 | 38.19 | 16.60 | 28.84 |
| SFT→GRPO | 1.3 | 1.8 | 3.1 | 79.49 | 96.90 | 56.23 | 83.33 | 66.31 | 37.96 | 18.80 | 29.94 |
| SFT→GRPO | 1.8 | 0.0 | 1.8 | 79.49 | 96.81 | 54.28 | 83.33 | 72.91 | 38.36 | 16.20 | 27.36 |
| SFT→GRPO | 1.8 | 0.4 | 2.2 | 79.93 | 97.42 | 56.19 | 83.33 | 66.58 | 38.56 | 17.60 | 29.21 |
| SFT→GRPO | 1.8 | 0.7 | 2.5 | 80.54 | 97.64 | 57.08 | 83.33 | 66.22 | 38.24 | 18.60 | 29.02 |
| SFT→GRPO | 1.8 | 1.1 | 2.9 | 81.07 | 97.73 | 57.25 | 83.33 | 65.33 | 38.07 | 18.40 | 28.47 |
| SFT→GRPO | 1.8 | 1.5 | 3.3 | 80.72 | 97.64 | 56.53 | 83.33 | 63.81 | 38.15 | 17.40 | 30.13 |
| SFT→GRPO | 2.2 | 0.0 | 2.2 | 81.33 | 97.68 | 56.19 | 83.33 | 66.40 | 38.04 | 18.40 | 29.02 |
| SFT→GRPO | 2.2 | 0.4 | 2.6 | 81.68 | 97.90 | 56.91 | 83.33 | 59.36 | 37.97 | 18.40 | 28.47 |
| SFT→GRPO | 2.2 | 0.7 | 2.9 | 80.94 | 97.20 | 56.40 | 83.33 | 69.96 | 38.10 | 16.00 | 28.47 |
| SFT→GRPO | 2.2 | 1.1 | 3.3 | 80.94 | 97.11 | 57.46 | 83.33 | 69.69 | 37.95 | 17.80 | 28.10 |
| SFT→GRPO | 2.7 | 0.0 | 2.7 | 81.29 | 97.68 | 56.02 | 83.33 | 67.47 | 38.44 | 18.40 | 28.10 |
| SFT→GRPO | 2.7 | 0.4 | 3.1 | 81.90 | **98.03** | 57.42 | 83.33 | 54.55 | 38.46 | 15.60 | 29.02 |

### B.1 Positive-only SFT control

Table 6: Complete checkpoint results for the positive-only SFT control, with the base model for comparison. Training uses xLAM examples whose reference answers contain at least one function call. Scores are percentages (↑: higher is better); bold marks the highest score in each metric column within this table, including ties and the base model.

| Recipe | Training time (h) | | | xLAM | | BFCL-v3 | | | General capabilities | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | SFT | GRPO | Total | ETC ↑ | FNA ↑ | Single-turn FC (agg) ↑ | Relevance ↑ | Irrelevance (agg) ↑ | MMLU ↑ | MATH-500 ↑ | IFEval ↑ |
| Base | 0.0 | 0.0 | 0.0 | 63.84 | 90.03 | 50.28 | 77.78 | **66.93** | **42.04** | 15.80 | 26.80 |
| SFT | 0.5 | 0.0 | 0.5 | 75.73 | 96.81 | 53.85 | 94.44 | 13.28 | 41.77 | 17.00 | 28.10 |
| SFT | 0.9 | 0.0 | 0.9 | 77.96 | 97.33 | 53.60 | 94.44 | 7.04 | 41.61 | **19.80** | 29.57 |
| SFT | 1.4 | 0.0 | 1.4 | 79.23 | 97.64 | 55.34 | **100.00** | 8.02 | 41.83 | 18.60 | 26.62 |
| SFT | 1.8 | 0.0 | 1.8 | 81.24 | 97.99 | **57.59** | **100.00** | 9.09 | 41.56 | 19.20 | **31.42** |
| SFT | 2.3 | 0.0 | 2.3 | 81.59 | 98.08 | 57.34 | **100.00** | 7.31 | 41.47 | 18.40 | 31.05 |
| SFT | 2.7 | 0.0 | 2.7 | **82.12** | **98.29** | 56.19 | **100.00** | 7.22 | 41.72 | 18.60 | 29.76 |