Extended Abstract

Motivation Inference-time compute scaling has emerged as a compelling alternative to parameter scaling for improving LLM performance. By generating multiple candidate responses and selecting the best, a smaller model can in principle match or exceed a much larger one. However, the effectiveness of this approach hinges on verifier quality — and prior work on high-quality verifiers typically relies on large models (7B parameters and above) and substantial human-labeled supervision, potentially negating the cost savings that motivate inference-time scaling in the first place. This work asks whether these requirements can be dramatically relaxed: *Can a 0.5B generative verifier, trained solely on rollouts already produced during policy training, provide meaningful best-of-N gains?* We investigate this question across three policy training frameworks — SFT, IPO, and RLOO — that differ in how strongly they optimize the output distribution, revealing that the choice of training framework has significant consequences for inference-time scaling far beyond single-sample accuracy.

Method We train a small generative verifier (Qwen2.5-0.5B) to predict a single "Yes" or "No" token conditioned on a Countdown arithmetic prompt and a candidate rollout, scoring each candidate as $P(\text{"Yes"})/(P(\text{"Yes"}) + P(\text{"No"}))$. Critically, no additional data collection is required: the verifier is fine-tuned exclusively on rollouts already generated during policy training, augmented with oracle correctness labels. We compare two initialization strategies — a **Base Verifier** (off-the-shelf Qwen2.5-0.5B) and an **On-Policy Verifier** (initialized from the trained policy checkpoint) — against oracle best-of-N, random, and majority baselines across sample budgets $K \in \{1, 4, 8, 16\}$. The key novelty lies in combining sub-1B scale, zero additional data, and a systematic analysis of how training framework shapes verifier utility.

Implementation All models are Qwen2.5-0.5B. Policies are trained from scratch under SFT (behavioral cloning), IPO (identity preference optimization), and RLOO (REINFORCE leave-one-out) on the Countdown arithmetic task. The verifier is fine-tuned with cross-entropy loss computed only on the final Yes/No token. Evaluation uses 16 rollouts per Countdown prompt. **Headline results:** the SFT-fine-tuned verifier reaches the oracle best-of-N ceiling at large K; the RLOO-fine-tuned verifier comes within 5% of it; and Base and On-Policy verifiers perform comparably across most settings, with the largest gap being 6% in RLOO at $K = 4$.

Discussion The central finding is counterintuitive: SFT, the weakest single-sample policy, produces the best verifier performance at large K. We attribute this to a *diversity dividend* — SFT's unoptimized output distribution preserves a wide spread of solution approaches, giving the verifier a richer and more discriminable candidate pool. RLOO's RL training concentrates probability mass on high-reward modes, collapsing diversity and limiting the oracle ceiling, while paradoxically making individual verification easier when a correct answer is present. IPO occupies a middle ground, exhibiting answer-level diversity comparable to SFT but apparently introducing subtler inter-sample correlations that impair discrimination. Both verifier and majority selection exhibit a non-monotonic headroom dip at $K = 8$ across all three frameworks, suggesting a structural inflection point where the oracle ceiling begins outpacing any selection strategy. Limitations include answer-level-only diversity metrics (which cannot capture solution-path variation), evaluation on a single task and verifier size, and an unresolved explanation for the $K = 8$ inflection.

Conclusion A 0.5B generative verifier trained only on existing policy rollouts can achieve near-oracle best-of-N accuracy at no additional data or model cost. Candidate pool diversity — shaped by the choice of training framework — is the primary determinant of both the oracle ceiling and the effectiveness of inference-time selection. For practitioners deploying sub-1B models, these findings suggest that training framework choices have consequences well beyond single-sample quality, with significant implications for the scalability and cost-effectiveness of inference-time compute allocation.

Small Generative Verifiers for Inference-Time Scaling Across RL Training Frameworks

Jui Khankari [*1] **Josh Delgadillo** [*1]

Abstract

A high-leverage way to improve LLM performance is by allocating more compute at inference time through best-of-N sampling. However, the effectiveness of this technique hinges on the ability to identify a correct response among N candidates. High-quality verifiers may require large models and substantial additional training data, potentially canceling out the cost savings of inference-time scaling. We explore whether a small (0.5B) generative verifier, fine-tuned only on existing rollouts collected during policy training, can provide meaningful test-time gains across three RL training frameworks: SFT, IPO, and RLOO. We find that the SFT-fine-tuned verifier achieves the theoretical best-of-N ceiling and the RLOO-fine-tuned verifier comes within 5% of it. We further analyze how verifier utility varies with training framework, candidate pool homogeneity, and sample budget k, revealing a surprising interaction between output diversity and verification quality.

1. Introduction

Inference-time compute scaling has emerged as a promising alternative to parameter scaling for improving language model performance. By generating multiple candidate responses and selecting among them, a smaller model can match or exceed the performance of a much larger one (Snell et al., 2024). For small sample sizes and relatively straightforward prompts/tasks, simple sampling methods (i.e. majority rules – choosing the most common answer) suffice and yield high accuracy. However, these techniques do not scale to larger sample sizes and yield diminishing returns from additional samples. At these magnitudes, more advanced strategies, such as tree search or training a large model (i.e. an LLM) as a discriminative verifier, or leveraging LLM's natural text understanding as a generative verifier, are necessary to continue to extract performance.

The literature exploring high-quality verifiers produces impressive results, but often uses large LLMs and detailed, human-labeled data, which may negate the cost savings of inference-time scaling. Additionally, prior work explores generative verifiers, using the embedded text understanding to reduce the size of the model and amount of data required. However, the verifier models used are often still in excess of 7 billion parameters. This work motivates the following research questions that push the frontier of architecture cheapness and data efficiency:

1. Can an inexpensive generative verifier, a 0.5B model, provide meaningful test-time gains?
2. Can an inexpensive generative verifier be trained only on existing rollouts produced during the original policy's training process?

We investigate this question across three policy training frameworks: supervised fine-tuning (SFT), identity preference optimization (IPO), and REINFORCE leave-one-out (RLOO). Each starts from a 0.5B Qwen2.5 model and performs self-improvement on the Countdown arithmetic task. Our main contributions are:

1. We demonstrate that a small generative verifier (Qwen2.5 0.5B) trained solely on existing rollouts matches the oracle best-of-N accuracy in the SFT setting and comes within 5% in the RLOO setting on the Countdown task.
2. We characterize how candidate pool diversity, shaped by the choice of training framework, determines both the oracle ceiling and the effectiveness of verification at scale.
3. We provide direct evidence that IPO's answer-level diversity conceals deeper arithmetic-strategy homogeneity: IPO's incorrect solutions share nearly identical operation fingerprints with its correct ones, substantially reducing the structural signal available to a verifier.

[*]Equal contribution [1]Department of Computer Science, Stanford University, Stanford, USA. Correspondence to: Jui Khankari <juik@stanford.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

2. Related Work

Recent work has established that test-time compute can be a viable and sometimes superior alternative to parameter scaling. Brown et al. (2024) show that across a variety of tasks, coverage (the percentage of tasks solved by the model) scales with the number of samples, often in a direct log-linear relationship. While this suggests powerful inference-time scaling laws, the authors acknowledge that simple sampling methods such as majority voting do not scale past several hundred samples. Thus, for tasks that are not automatically verifiable, harnessing additional inference-time performance requires a nontrivial verifier. Snell et al. (2024) achieved similar results, demonstrating that a model leveraging inference-time compute can outperform a $14\times$ larger model in some cases, but that verifier quality is the key bottleneck for effective best-of-N search. The authors also note that the difficulty of generating and verifying responses is highly dependent on the prompt, suggesting there exists a "compute-optimal" strategy for allocating compute time between training (a larger base model) and inference time (a larger verifier model) for a given prompt. Wu et al. 2025 explores this further through an empirical analysis of the Pareto curve for different configurations of LLM size and verification strategy. The authors found that small LLMs and advanced verification strategies (weighted tree search) consistently outperformed larger models.

Generative verifiers (i.e., an LLM) have shown promise as one such "advanced verification strategy" allowing for detailed reward modeling to extract near best-of-N oracle sampling performance. Specifically, Zhang et al. (2025) obtains high-quality verifiers by treating verification as next-token prediction, leveraging the model's existing knowledge base rather than training a pure discriminative classifier. However, their basic generative verifier LLM only marginally improves on the LLM-as-a-judge baseline (prompting an off-the-shelf LLM, rather than fine-tuning one to predict Yes/No); the majority of their improvements come from a Generative+CoT verifier process. Cobbe et al. (2021) also achieves strong results on math reasoning tasks similar to Countdown but uses 6B and 175B models with human-labeled supervision. Perhaps the nearest-neighbor study to our experiments is Hosseini et al. (2024)'s V-Star framework. The authors leverage the existing correct and incorrect rollouts to train an LLM verifier using DPO. They find that this framework allows Llama-7B to outperform Llama-70B on an 8th-grade math dataset.

Considering the above corpus, it is clear that inference-time compute scaling is an emerging performance technique, and within it, generative verifiers have proven more effective than parameter scaling in some scenarios, especially in similar natural language math reasoning tasks. However, many of these studies still leverage fairly large models (7B and above) and only 1 trains solely on existing data collected during the policy training process, leaving open the question of whether similar gains are achievable for sub-1B models. Our work addresses this gap directly.

3. Methods

Each of our experimental configurations involved two distinct models:

1. The **policy** generates the responses to the Countdown task prompt
2. The **verifier** predicts a single token– "Yes" or "No" – conditioned on the Countdown task prompt and a rollout (generated by the policy).

In each experimental configuration (SFT, IPO, or RLOO), we perform the following process:

1. Train the policy (starting from an off-the-shelf Qwen2.5-0.5B model) according to the chosen RL framework (SFT, IPO, RLOO).
2. Train 2 different verifiers according to the process in Section 3.1.
   (a) The **Base Verifier** begins training from an off-the-shelf Qwen2.5-0.5B model.
   (b) The **On-Policy Verifier** beings training from the policy's training checkpoint produced in (1).
3. Evaluate the policy by generating 16 rollouts per Countdown prompt in the evaluation dataset.
4. Score each of the 16 rollouts in each Countdown prompt in the evaluation dataset using the verifier. The verifier outputs the log-probabilities of its first token, and the score is the softmax between "Yes" and "No":

$$\text{verifier\_score} = \frac{P(\text{“Yes”})}{P(\text{“Yes”}) + P(\text{“No”})}$$

5. Compute the evaluation metrics outlined in Section 3.2

3.1. Verifier Training

For each training framework {SFT, IPO, RLOO}, we:

1. Train the original policy and save the rollouts produced during training.
2. Augment each rollout into a verification prompt of the form:

```
{Countdown prompt}
Candidate solution:
{full rollout text}
Is this correct? {Yes/No --
determined by oracle}
```

3. Train the verifier. We perform supervised fine-tuning on either (1) an off-the-shelf Qwen2.5-0.5B model for the Base Verifier, or (2) the trained policy checkpoint for the On-Policy Verifier, using the augmented rollouts from (2) and cross-entropy loss computed only on the final `yes`/`no` token. With this loss setup, the verifier learns to predict only "Yes" or "No" conditioned on the Countdown prompt and candidate solution.

3.2. Evaluation

For $k \in \{1, 4, 8, 16\}$, we evaluate five selection strategies on each policy's generated candidates:

- **best-of-N@K (oracle)**: selects the correct answer if any exists among the first K samples, representing the theoretical ceiling for any selection strategy given the candidate pool (each selection strategy@K also has access only to the first K samples).
- **pass@K (standard)**: Over every possible size-k sample of the 16 rollouts, the probability that a given size-k sample has at least 1 correct rollout. Since pass@K sees all 16 rollouts, it is a measure of overall sample quality, rather than the ceiling for each sampling strategy.
- **random@K**: selects one response uniformly at random from K candidates.
- **majority@K**: selects the most frequently occurring answer among K candidates.
- **verifier@K**: selects the response assigned the highest probability of correctness (score) by the generative verifier.

Headroom is defined as oracle@K − strategy@K, measuring how far each selection strategy falls from the theoretical ceiling. A lower Headroom indicates a strategy gets closer to fully exploiting the available candidate pool.

3.3. Solution-Level Strategy Analysis

To probe the structural diversity of rollouts beyond answer-level metrics, we define each solution's *arithmetic strategy fingerprint* as the sorted multiset of operators ($+$, $-$, $\times$, $\div$) appearing in its final `<answer>` equation. This representation captures the combination of operations used to reach a solution, independent of the specific numbers involved. For each framework we compute:

- **Within-prompt unique fingerprint ratio**: for each prompt's K rollouts, the fraction of rollouts with a distinct fingerprint.
- **Global fingerprint entropy**: Shannon entropy (nats) over the fingerprint distribution across all rollouts.
- **Correct/incorrect fingerprint overlap**: per-prompt Jaccard similarity between the fingerprint sets of correct and incorrect rollouts. High overlap means wrong answers are structurally indistinguishable from right ones, reducing the signal available to a verifier.
- **Mul/div usage rate**: fraction of rollouts using at least one $\times$ or $\div$ operator, a proxy for higher-complexity strategy usage.

4. Results

4.1. Verifier Accuracy Across Training Frameworks

The SFT-fine-tuned verifier achieves best-of-N accuracy at the theoretical oracle ceiling, while the RLOO-fine-tuned verifier comes within 5% of the ceiling. Counterintuitively, the RLOO verifier achieves lower overall accuracy than the IPO verifier despite RLOO being the stronger single-sample policy (highest pass@K at $K = 1$, Figure 1).

The headroom of both verifier and majority selection decreases from $K = 4$ to $K = 8$, then increases from $K = 8$ to $K = 16$ (Figure 2), consistent across all three training methods. This non-monotonic behavior suggests the oracle ceiling grows faster than either selection strategy can track beyond $K = 8$ — a structural inflection point where larger candidate pools uniformly challenge selection quality.

4.2. Effect of Candidate Diversity

Figure 3 shows that SFT maintains the highest unique answer ratio and normalized entropy across all K values. Since SFT does not optimize toward any reward signal, it preserves the full spread of the model's output distribution: as K grows, the probability that at least one correct solution exists in the pool increases substantially, giving the verifier more signal to work with. RLOO collapses to more homogeneous distributions at larger K, consistent with RL training concentrating probability mass on high-reward modes — limiting the oracle ceiling even as sample budget grows.

4.3. Verifier Headroom Reduction by Framework

The generative verifier provides a larger reduction in headroom for RLOO than for IPO. When RLOO generates K candidates, their homogeneity means a correct answer — when it exists — stands out more distinctly against a background of similar incorrect answers, giving the verifier a cleaner discrimination problem: "which of these nearly identical candidates is actually correct?" For IPO, the more diverse candidate pool poses a harder discrimination problem, limiting relative headroom reduction. However,

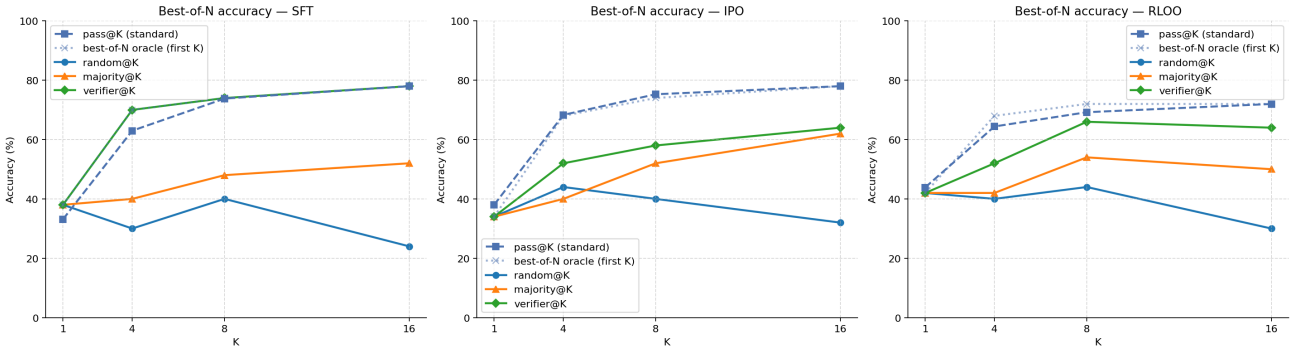

Figure 1. Best-of-N accuracy as a function of sample size K for SFT, IPO, and RLOO. Each panel compares pass@K (standard), best-of-N oracle, random@K, majority@K, and verifier@K. Notably, the oracle upper bound for SFT and IPO surpasses RLOO at $K \geq 8$, despite RLOO achieving the highest single-sample accuracy at $K = 1$ — a reversal driven by RLOO's reduced output diversity at larger sample budgets.

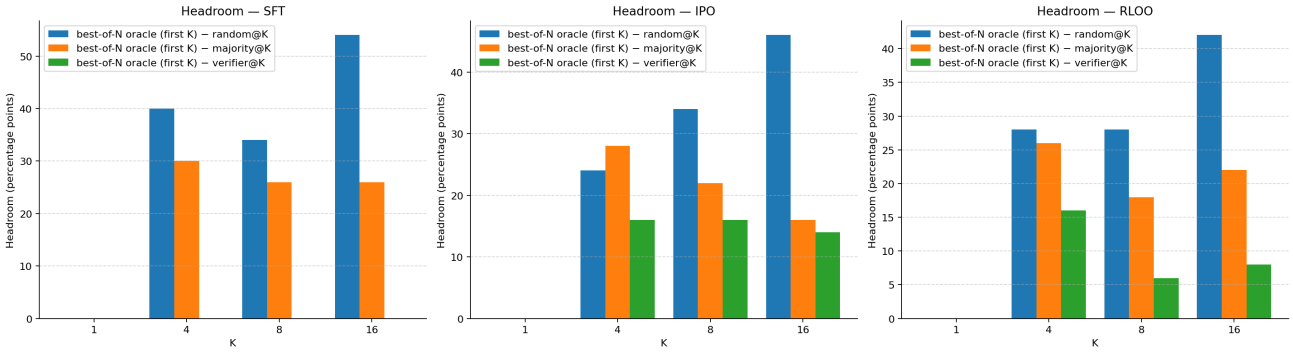

Figure 2. Headroom (oracle@K − strategy@K) across training methods. Larger positive values indicate a strategy is further from the theoretical ceiling. Headroom grows with K for all methods, and exhibits a non-monotonic dip at $K = 8$ for verifier and majority selection — consistently across all three training frameworks — before increasing again at $K = 16$.

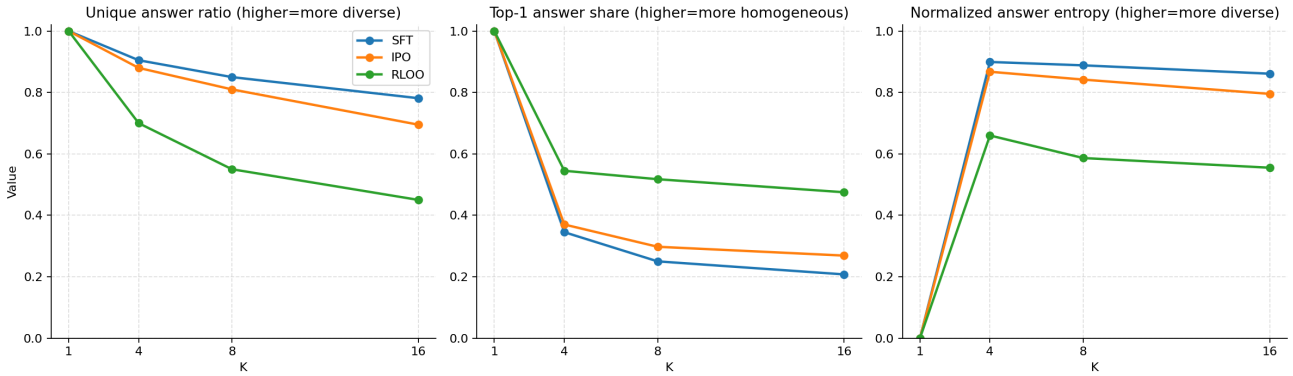

Figure 3. Output diversity metrics — unique answer ratio (higher = more diverse), top-1 answer share (higher = more homogeneous), and normalized answer entropy (higher = more diverse) — as a function of K for SFT, IPO, and RLOO. SFT maintains the highest diversity across all metrics; RLOO collapses toward more homogeneous distributions at larger K, while SFT and IPO exhibit nearly identical diversity profiles despite differing verifier performance.

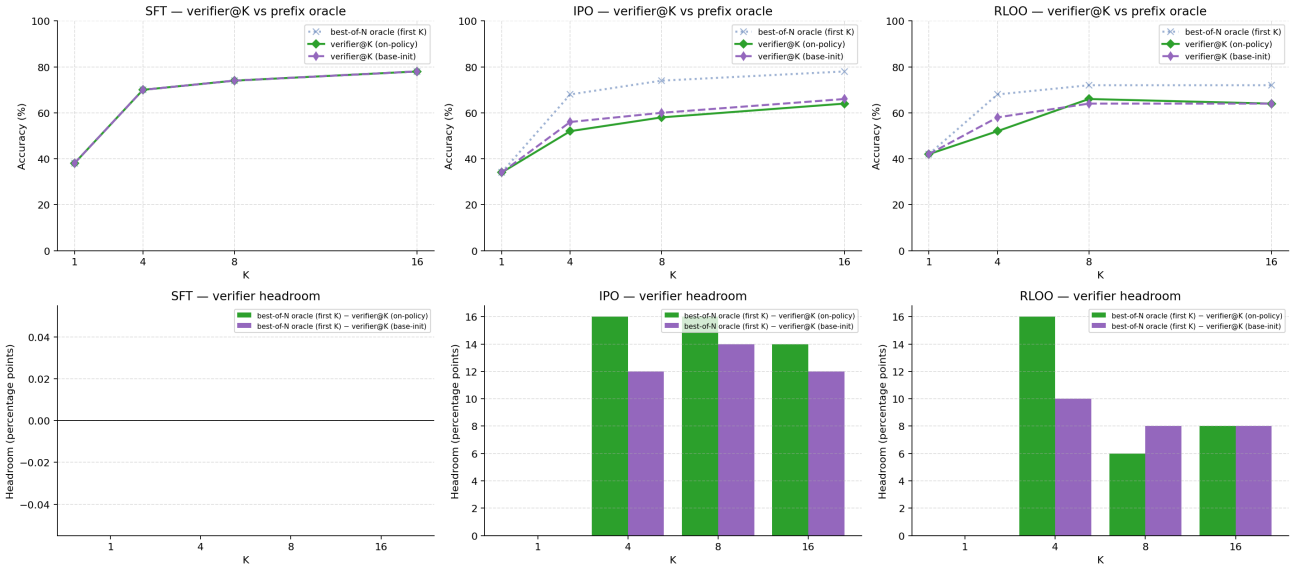

Figure 4. A comparison between the Base and On-Policy verifiers across SFT, IPO, and RLOO. In all settings, each verifier performs comparably, with an outlier significant difference of 6% in RLOO at k=4. For SFT, both verifiers recover oracle performance, while in the IPO setting, the Base Verifier performs better than the On-Policy verifier. In RLOO, the Base Verifier is superior for $k \leq 8$ and the On-Policy verifier is superior otherwise.

IPO's higher oracle ceiling means absolute accuracy remains higher than RLOO. The SFT verifier benefits from both a rich, diverse pool and sufficient per-sample quality, reaching the oracle ceiling: even imperfect discrimination over a rich pool beats perfect discrimination over a poor one.

4.4. Solution-Level Strategy Diversity

Although SFT and IPO exhibit nearly identical answer-level diversity (Figure 3), we find that they diverge substantially at the level of arithmetic strategy. Figure 5 summarizes four fingerprint-based metrics computed across 50 prompts $\times$ 16 rollouts per framework.

Global and within-prompt strategy entropy. SFT produces a more diverse distribution of operation fingerprints: global fingerprint entropy is 2.84 nats for SFT versus 2.52 for IPO, and the within-prompt unique fingerprint ratio is 0.414 versus 0.368. SFT also uses multiplication or division in 34.5% of rollouts compared to 25.8% for IPO, indicating IPO converges toward simpler additive and subtractive strategies. First-operation distributions are nearly identical between the two frameworks (both favor subtraction as the first operator), ruling out the opening move as the source of divergence; the difference lies in the full operation sequence.

Correct/incorrect structural overlap. The most diagnostic metric is the Jaccard similarity between the fingerprint sets of correct and incorrect rollouts within each prompt (Figure 6). For IPO, correct and incorrect answers share a mean Jaccard overlap of 0.179 — nearly $3\times$ higher than SFT's 0.064. This means that for the majority of prompts, IPO's wrong answers use the same combination of arithmetic operations as its right ones. The verifier therefore receives no structural discrimination signal from the operation fingerprint: a correct and an incorrect IPO response look the same at the strategy level. For SFT, incorrect solutions are structurally more distinct from correct ones, giving the verifier a meaningful surface to discriminate over.

These findings provide direct, mechanism-level evidence for the hypothesis raised in the Discussion: IPO's preference optimization collapses strategy diversity even while preserving answer diversity, and this collapse is precisely what limits verifier utility at large K.

5. Discussion

5.1. Base Verifier Outperforms On-Policy Initialization in IPO and Low-K RLOO

Despite the On-Policy Verifier sharing weights with the trained policy and therefore having direct exposure to its output distribution, the Base Verifier matches or exceeds it in the IPO setting and in RLOO at $k \leq 8$. We attribute this to representation interference: the policy checkpoint has been fine-tuned to generate high-reward solutions, which may bias its internal representations toward producing output rather than evaluating it. When this checkpoint is subsequently fine-tuned as a verifier, the optimization must partially overwrite generation-oriented features, potentially

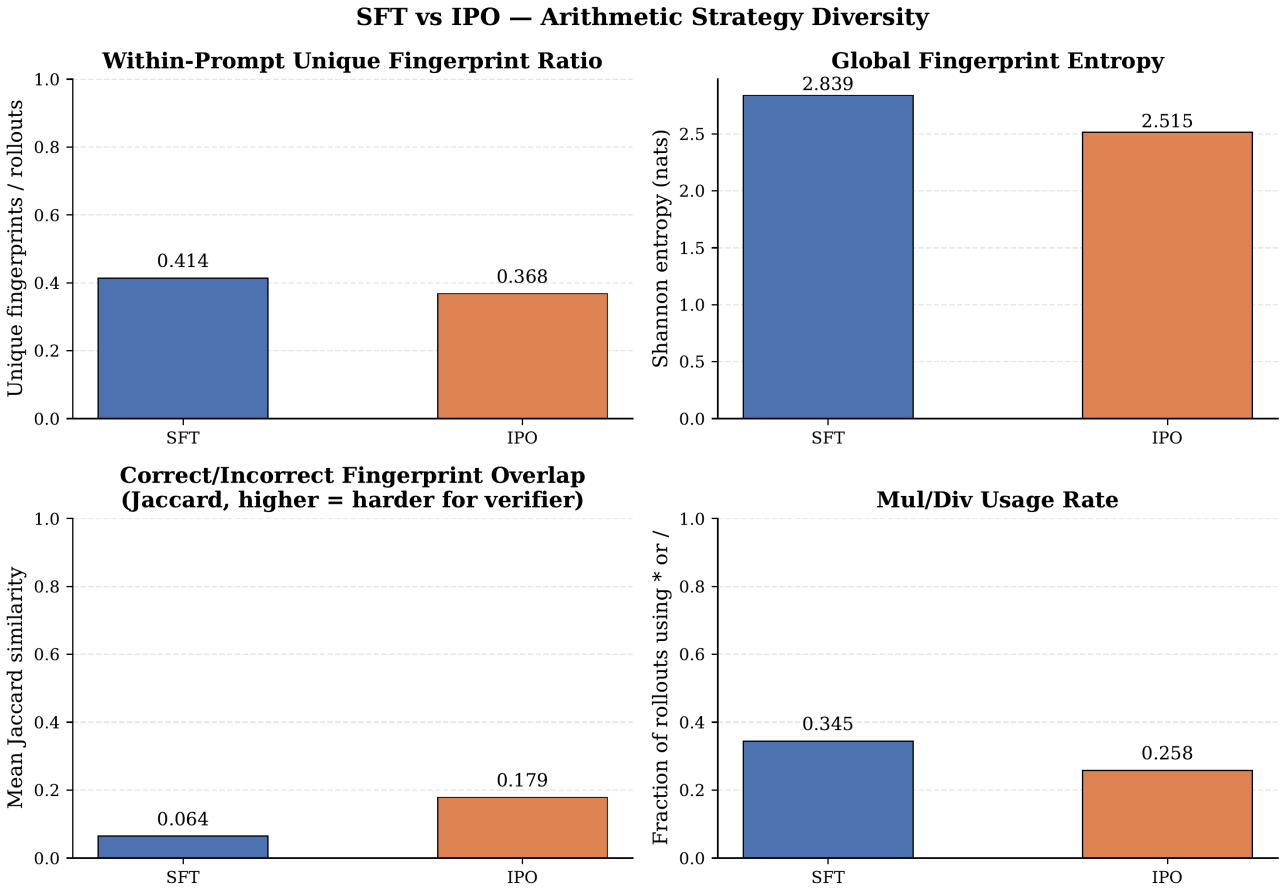

Figure 5. Arithmetic strategy diversity metrics for SFT and IPO. **Top left**: within-prompt unique fingerprint ratio (fraction of a prompt's 16 rollouts with a distinct operation multiset). **Top right**: global fingerprint entropy (nats) over all rollouts. **Bottom left**: mean per-prompt Jaccard similarity between the fingerprint sets of correct and incorrect rollouts — higher values indicate wrong answers are structurally indistinguishable from correct ones, reducing verifier discriminability. **Bottom right**: fraction of rollouts using at least one multiplication or division operator. SFT exhibits higher strategy entropy and lower correct/incorrect structural overlap, despite near-identical answer-level diversity (Figure 3).

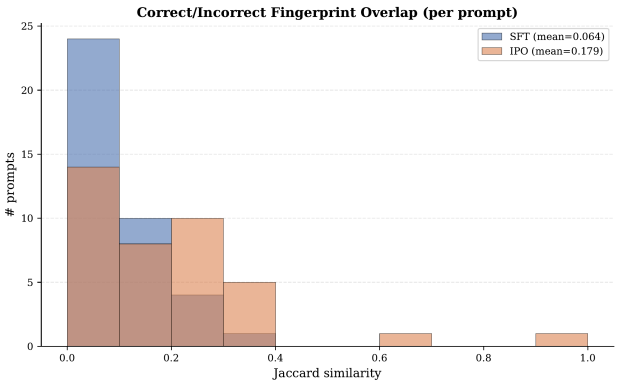

Figure 6. Per-prompt distribution of Jaccard similarity between the fingerprint sets of correct and incorrect rollouts, for SFT (blue) and IPO (orange). IPO's distribution is shifted right (mean $= 0.179$) relative to SFT (mean $= 0.064$): for the majority of prompts, IPO's incorrect solutions share operation fingerprints with its correct ones, leaving the verifier without structural discrimination signal.

degrading the neutral, comparative representations that verification requires. The off-the-shelf base model, by contrast, begins from a more general-purpose feature space and can specialize toward discrimination without fighting prior generative biases.

This effect is most pronounced in IPO, where preference optimization has actively shaped the model's representations to favor certain solution styles, precisely the kind of distributional skew that would impair unbiased verification. In RLOO, the same interference appears at small k, where the candidate pool is too homogeneous for the On-Policy Verifier's familiarity with the distribution to compensate for its representational bias. At $k = 16$, the On-Policy Verifier recovers: a richer candidate pool may provide enough contrastive signal for the shared policy representations to become an asset rather than a liability.

Why SFT outperforms IPO at large K despite lower per-sample quality. At $K = 1$, IPO's preference optimization yields higher per-sample accuracy than SFT's behavioral cloning (evidenced by IPO's higher $K = 1$ accuracy in Figure 1), indicating higher individual solution quality. Yet SFT's verifier outperforms IPO's at large K, despite both exhibiting nearly identical answer-level diversity (Figure 3). Our solution-level analysis (Section 4.4) provides direct evidence for why: IPO's incorrect solutions share nearly identical operation fingerprints with its correct ones (mean Jaccard overlap 0.179 vs. 0.064 for SFT), eliminating the structural signal the verifier relies on to discriminate. IPO's optimization pushes all solutions — correct and incorrect alike — toward a narrow range of arithmetic strategies, while SFT's unoptimized distribution covers a wider set of approaches. The resulting structural gap between correct and incorrect solutions is far larger for SFT, making the verifier's discrimination task substantially easier. The diversity dividend from SFT ultimately outweighs IPO's per-sample quality advantage.

RLOO homogeneity as a double-edged sword. RLOO's low diversity limits the oracle ceiling, since the correct solution may simply not exist in a homogeneous candidate pool. At the same time, when a correct answer is present, its distinctness from the homogeneous background makes verification easier — explaining the larger relative headroom reduction from the RLOO verifier compared to IPO. The same property that makes RLOO a weaker base model for scaling K makes it a more verifier-friendly distribution for the candidates it does generate.

6. Limitations and Future Work

Our results reveal two phenomena that the current analysis partially but not fully explains. First, we show that IPO's answer-level diversity conceals deeper arithmetic-strategy homogeneity (Section 4.4): IPO's correct and incorrect solutions share nearly identical operation fingerprints, reducing verifier discriminability. However, our fingerprint analysis captures only the multiset of operators in the final equation — it cannot determine whether solutions with the same fingerprint took different reasoning paths to arrive at the equation, or whether the verifier's difficulty stems from reasoning-chain similarity rather than strategy similarity. Extending this analysis to the full `<think>` chain (e.g., counting the number of candidate equations explored, or embedding reasoning chains for clustering) would further sharpen the mechanistic account. Second, headroom for both verifier and majority selection non-monotonically dips at $K = 8$ before rising again at $K = 16$, consistently across all three training methods — suggesting something structural about how candidate pools evolve at larger sample budgets that uniformly challenges selection strategies, rather than a method-specific artifact. Understanding this inflection point could inform more principled sample budget selection.

These findings are subject to several remaining limitations. We evaluate on a single task domain (Countdown arithmetic) and a single verifier size (0.5B), so it remains unclear whether the relative performance ordering of SFT, IPO, and RLOO would hold across different problem types or with a larger verifier. The structural homogeneity we observe in IPO may also be task-specific: on tasks with richer solution spaces, IPO's preference optimization might preserve more strategy diversity.

Future work should extend the solution-type clustering beyond operator fingerprints to include intermediate-value sequences and reasoning-chain structure, and test whether

the SFT verifier advantage generalizes to other arithmetic or logical reasoning tasks.

7. Conclusion

We demonstrate that a small (0.5B) generative verifier, trained solely on rollouts already produced during policy training, can achieve near-oracle best-of-N accuracy without any additional human labeling or model scale. The SFT-fine-tuned verifier reaches the theoretical ceiling; the RLOO-fine-tuned verifier comes within 5%. Counterintuitively, SFT — the weakest single-sample policy — produces the best verifier performance at large K, driven by the diversity dividend of its unoptimized output distribution. Our solution-level analysis reveals the mechanism: IPO's preference optimization collapses arithmetic strategy diversity even while preserving answer-level diversity, causing its incorrect solutions to share operation fingerprints with correct ones (Jaccard 0.179 vs. 0.064 for SFT) and eliminating the structural signal the verifier relies on. Our analysis reveals that candidate pool diversity, shaped by the choice of training framework, is the primary determinant of both the oracle ceiling and the effectiveness of inference-time selection strategies. These findings suggest that for sub-1B models, the training framework has implications well beyond single-sample quality, with significant consequences for the scalability and cost-effectiveness of inference-time compute allocation.

8. Team Contributions

Both members of the team contributed equally to the project.

References

Brown, B., Juravsky, J., Ehrlich, R., Clark, R., Le, Q. V., Ré, C., and Mirhoseini, A. Large language monkeys: Scaling inference compute with repeated sampling, 2024. URL https://arxiv.org/abs/2407.21787.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., and Schulman, J. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Hosseini, A., Yuan, X., Malkin, N., Courville, A., Sordoni, A., and Agarwal, R. V-star: Training verifiers for self-taught reasoners, 2024. URL https://arxiv.org/abs/2402.06457.

Snell, C., Lee, J., Xu, K., and Kumar, A. Scaling llm test-time compute optimally can be more effective than scaling model parameters, 2024. URL https://arxiv.org/abs/2408.03314.

Wu, Y., Sun, Z., Li, S., Welleck, S., and Yang, Y. Inference scaling laws: An empirical analysis of compute-optimal inference for problem-solving with language models, 2025. URL https://arxiv.org/abs/2408.00724.

Zhang, L., Hosseini, A., Bansal, H., Kazemi, M., Kumar, A., and Agarwal, R. Generative verifiers: Reward modeling as next-token prediction, 2025. URL https://arxiv.org/abs/2408.15240.