Evolution Strategies at Scale: LLM Fine-Tuning Beyond Reinforcement Learning

Xin Qiu*†[1], Yulu Gan*[2], Conor F. Hayes*[1], Qiyao Liang[2], Yinggan Xu[3], Roberto Dailey[1], Elliot Meyerson[1], Babak Hodjat[1], Risto Miikkulainen[1,4]
[1]Cognizant AI Lab, [2]MIT, [3]UCLA, [4]UT Austin, *Equal Contribution, †Project Lead

1. Introduction

LLM Fine-tuning: Fine-tuning is a standard step for adapting pre-trained models to downstream tasks and aligning be- havior with user preferences. In practice, reinforcement learning (RL) has become the pre- dominant choice for this fine-tuning stage.

Limitations of RL fine-tuning methods:

- Difficulty with long-horizon reward
  - Credit assignment at token level is challenging
- High variance in performance
  - Instability across models
- Reward hacking
  - Triggering undesirable behaviors
- Sensitivity to hyperparameter setup
  - Increasing fine-tuning cost

Evolution Strategies (ES):

- Population-based zeroth-order optimization
- Never been applied to billion-parameter space

Advantages:

- Only needs response-level rewards
- Backpropagation-free, no gradient calculation
- Small population (e.g., 30) works when searching in billion-parameter space
- Robust across different model sizes/families
- Less tendency to reward hacking
- Insensitivity to hyperparameter setup
- Stability across runs
- Embarrassingly parallelizable

2. Method

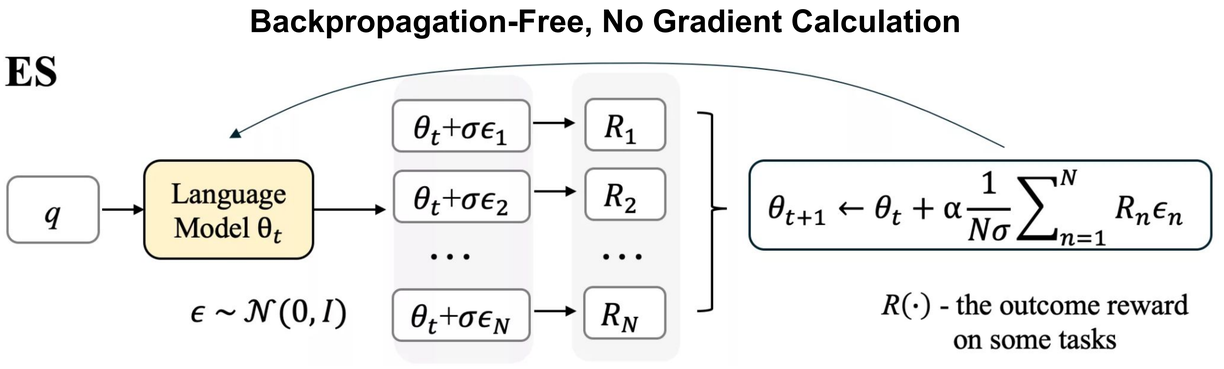

Figure 1: Process of LLM fine-tuning with Evolution Strategies (ES). Given a pretrained LLM with initial parameters θ_0 and a target reward function $R(\cdot)$, the task is to fine-tune the parameters so that the reward function is optimized. In each iteration, N perturbed models are sampled by adding random Gaussian noise ε_n to their parameters. The noise is i.i.d. in each dimension of the parameter space, and it is scaled by the hyperparameter σ. The perturbed models are evaluated to obtain their reward scores R_n. The final update of the model parameters aggregates the sampled perturbations by weighting them using their normalized reward scores. The updated model parameters serve as the base for the next iteration, and the same process repeats.

Scalability Enhancements:

- Noise retrieval with random seeds
  - No storage of entire weights
- Parallel evaluations
  - Wall-clock time speed-up
- In-place perturbation and restoration
  - Inference-level GPU memory
- Decomposed parameter update
  - Significant reduction of peak GPU memory

"ES-at-Scale" Library:

- Open-source, free to use
- Inference engine acceleration
- Multi-GPU distributed execution
- Inference-level hardware requirement
- Easily adaptable to any models/tasks

ES-at-Scale

3. Empirical Studies

The fine-tuning performance of ES is compared with RL baselines on a standard reasoning benchmark, i.e., the countdown task. After that, behavioral differences between ES and RL are investigated in fine-tuning for conciseness, followed by comparisons to more SOTA RL baselines on several math reasoning tasks. Finally, ES is applied to solve two challenging puzzle problems.

Base Model	Original	RL					ES (ours)
		PPO-z	GRPO-z (8)	GRPO-z (30)	GRPO-v	Dr.GRPO-v	
Qwen-2.5-0.5B-Instruct	0.1	0.3	0.3	0.5	13.0	13.5	**14.4**
Qwen-2.5-1.5B-Instruct	0.7	14.2	13.9	14.8	27.8	31.0	**37.3**
Qwen-2.5-3B-Instruct	10.0	20.1	30.9	32.5	37.8	43.8	**60.5**
Qwen-2.5-7B-Instruct	31.2	55.1	54.2	52.8	57.0	57.5	**66.8**
Llama-3.2-1B-Instruct	0.4	11.2	14.5	13.0	14.9	13.9	**16.8**
Llama-3.2-3B-Instruct	3.2	35.3	39.4	38.8	42.5	47.8	**51.6**
Llama-3.1-8B-Instruct	8.1	42.8	49.9	51.3	46.9	50.2	**61.2**

Figure 2: Accuracy (%) on the Countdown task across model families, sizes, and fine-tuning algorithms. The same hyperparameters were used for all ES runs; a separate grid search for the best hyperparameters was run for each RL experiment. ES achieves consistently strong performances across different models sizes and families.

Task	Base Model	Original	ES Fine-tuned
ARC-AGI	Qwen-2.5-14B	0.2	**29.5**
Sudoku	Qwen-2.5-3B	2.5	**69.5**

Figure 3: Accuracy (%) on solving puzzle problems. *Original* refers to directly evaluating the base model without any fine-tuning. ES fine-tuning improves performance significantly, demonstrating that it can be applied to a range of problems.

Model	β	α	σ	Reward ↑	KL ↓
Qwen-2.5-7B+GRPO	0.0	5×10^{-6}	✗	$0.867 \pm 0.054^*$	$0.861 \pm 0.614^*$
Qwen-2.5-7B+GRPO	0.01	5×10^{-6}	✗	$0.871 \pm 0.060^*$	$1.354 \pm 0.873^*$
Qwen-2.5-7B+GRPO	0.0167	5×10^{-6}	✗	0.911 ± 0.038	1.591 ± 0.811
Qwen-2.5-7B+GRPO	0.0464	5×10^{-6}	✗	0.881 ± 0.062	1.384 ± 1.187
Qwen-2.5-7B+ES	✗	0.0005	0.001	$0.889 \pm \mathbf{0.004}$	$0.274 \pm \mathbf{0.096}$
Qwen-2.5-7B+ES	✗	0.00075	0.0015	$0.919 \pm \mathbf{0.008}$	$0.813 \pm \mathbf{0.212}$

Figure 4: Behavior or GRPO and ES in terms of mean conciseness reward and mean KL divergence. The label * indicates cases where reward hacking was observed. ES exhibited lower performance variances across runs.

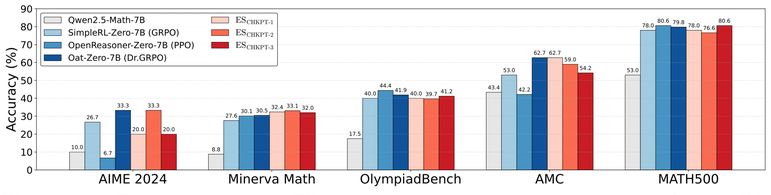

Figure 5: Performance of ES compared to strong, well-established RL baselines across math reasoning benchmarks. Across all benchmarks, ES achieved competitive performance. Given the vanilla nature of the current ES implementation, these results constitute a promising starting point for ES fine-tuning in math.

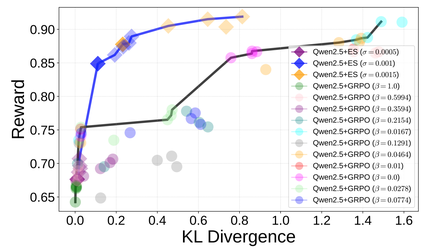

Figure 6: Mean conciseness reward and mean KL divergence from the base model for each fine-tuning checkpoint across different learning parameters. The Pareto front of ES (blue line) is higher and to the left of the GRPO Pareto front (black line) models, indicating that it found better tradeoffs. ES discovers these solutions without any KL divergence penalty, suggesting that it represents a distinctly different fine-tuning mechanism from the GRPO.

4. What's Next

- Blessing of Dimensionality
  - More parameters are easier to optimize
- Extension to Quantized Models
  - Handling non-differentiable discrete space
- Optimizing Model-level rewards
  - Optimizing for metacognitive alignment
- Effect of Weight-space Random Walks
  - Overcoming forgetting in continuous learning
- Combining ES with RL
  - ES for exploration and RL for exploitation
- Dedicated Infrastructure
  - Faster inference is all you need

5. Source Code & Contact

Code: **github.com/VsonicV/es-fine-tuning-paper** Correspondence: **Xin Qiu <qiuxin.nju@gmail.com>**